

IDENTIFICATION

PRODUCT ID: ZZ-ESKAA-10.1

PRODUCT TITLE: EVSAA- VAX 11/780 LOCAL CONSOLE STANDARD VERSION

DECO/DEPO: 10.1

DATE: MARCH 1986

MAINTAINED BY: VAX DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE
CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH
IS NOT SUPPLIED BY DEC.

ZZ-ESKAA-10.1 Document

!VAX-11/780 CONSOLE HELP FILE REV. 8 March 29, 1982

!TO STOP PRINTING, TYPE ^C

!FOR ABBREVIATION RULES, TYPE 'aABBREV.HLP'

!FOR ERROR MESSAGE HELP, TYPE 'aERROR.HLP'

!FOR REMOTE ACCESS HELP, TYPE 'aREMOTE.HLP'

!FOR WCS MICRO-DEBUGGER HELP, INSERT WCS DEBUG FLOPPY THEN TYPE 'aWCSMON.HLP'

!SYNTAX: ALL COMMANDS ARE TERMINATED BY CARRIAGE RETURN.

'EXAMINE' AND 'DEPOSIT' <QUAL> SWITCHES FOR ADDRESS SPACE :

'/P' = PHYSICAL MEMORY (THE DEFAULT)

'/V' = VIRTUAL MEMORY

'/I' = INTERNAL (PROCESSOR) REGISTERS

'/G' = GENERAL REGISTERS 0 THRU F (R0 THRU PC)

'/VB' = VBUS REGISTERS

'/ID' = IDBUS REGISTERS

'EXAMINE' AND 'DEPOSIT' <QUAL> SWITCHES FOR DATA-LENGTH :

'/B' = BYTE (8 BITS)

'/W' = WORD (2 BYTES)

'/L' = LONGWORD (2 WORDS)

'/Q' = QUADWORD (4 WORDS)

<ADDR> IS A <NUMBER>, OR ONE OF THE FOLLOWING SYMBOLIC ADDRESS

'R0,R1,R2,...,R11,AP,FP,SP,PC' (GENERAL REGISTERS)

'PSL' = PROCESSOR STATUS WORD

'*' = LAST ADDRESS

'+' = ADDRESS FOLLOWING 'LAST' (*) ADDRESS

'-' = ADDRESS PRECEDING 'LAST' (*) ADDRESS

'a' = USES LAST EXAMINE/DEPOSIT DATA FOR ADDRESS

<NUMBER> = STRING OF DIGITS IN CURRENT DEFAULT RADIX,

OR STRING OF DIGITS PREFIXED WITH A DEFAULT RADIX

OVERRIDE (x0 FOR OCTAL, xX FOR HEX).

'BOOT'

'BOOT <DEVNAM>'

-BOOTS THE CPU FROM DEFAULT DEVICE
 -TAKES THE FIRST THREE ALPHANUMERIC
 CHARS OF <DEVNAM>, AND EXECUTES THE
 INDIRECT FILE '<DEVNAM>BOO.CMD'

'CLEAR STEP'

'CLEAR SOMM'

-ENABLE NORMAL (NO STEP) MODE
 -CLEAR 'STOP ON MICRO-MATCH' ENABLE.
 NOTE: ID REGISTER 21 IS THE
 MICRO-MATCH REGISTER.

'CONTINUE'

'DEPOSIT[/<SWITCH(ES)>] <ADDR> <DATA>'

'ENABLE DX1:'

-ISSUES A CONTINUE TO THE ISP
 -DEPOSIT <DATA> TO <ADDRESS>
 -ENABLES CONSOLE SOFTWARE TO ACCESS
 FLOPPY DRIVE 1 ON THOSE SYSTEMS WITH
 DUAL FLOPPY

'EXAMINE[/<SWITCH(ES)>] <ADDR>'

'EXAMINE IR'

-DISPLAY COMMENTS OF <ADDRESS>
 -EXAMINE INSTRUCTION REG (IR). DISPLAYS
 OP-CODE, SP, IR, EXECUTION POINT
 COUNTER

'HALT'

'HELP'

'INITIALIZE'

'LINK'

-HALTS THE ISP
 -PRINTS THIS FILE
 -INITIALIZES THE CPU
 -CAUSES CONSOLE TO BEGIN COMMAND
 LINKING. CONSOLE PRINTS REVERSED
 PROMPT TO INDICATE LINKING. ALL
 COMMANDS TYPED BY USER WHILE LINKING
 ARE STORED IN AN INDIRECT COMMAND
 FILE FOR LATER EXECUTION. CONTROL-C
 TERMINATES LINKING. (SEE PERFORM)

'LOAD[/START:<ADDR>] <FILENAME>'

'LOAD/WCS <FILENAME>'

'NEXT <NUMBER>'

'PERFORM'

'QCLEAR <ADDRESS>'

'REBOOT'

'REPEAT <ANY-CONSOLE-COMMAND>'

'SET CLOCK SLOW'

'SET CLOCK FAST'

'SET CLOCK NORMAL'

'SET DEFAULT <OPTION>[,...,<OPTION>]'

'SET RELOCATION:<NUMBER>'

'SET SOMM'

'SET STEP BUS'

'SET STEP INSTRUCTION'

'SET STEP STATE'

'SET TERMINAL FILL:<NUMBER>'

'SET TERMINAL PROGRAM'

'SHOW'

'SHOW VERSION'

'START <ADDRESS>'

'TEST'

'TEST/COM'

'UNJAM'

'WCS'

'WAIT DONE'

-LOAD FILE TO MAIN MEMORY, STARTING AT ADDRESS 0, OR <ADDR> IF SPECIFIED

-LOAD FILE SPECIFIED TO WCS

-<NUMBER> STEP CYCLES ARE DONE, TYPE OF STEP DEPENDS ON LAST 'SET STEP' COMMAND

-EXECUTE A FILE OF LINKED COMMANDS PREVIOUSLY GENERATED VIA A 'LINK' COMMAND.

-DOES A QUAD CLEAR TO <ADDRESS>, WHICH IS FORCED TO A QUAD WORD BOUNDARY. (CLEARS ECC ERRORS)

-CAUSES A CONSOLE SOFTWARE RELOAD

-CAUSES THE CONSOLE TO REPEATEDLY EXECUTE THE <CONSOLE-COMMAND>, UNTIL STOPPED BY A CONTROL-C (^C).

-SET CPU CLOCK FREQ TO SLOW.

-SET CPU CLOCK FREQ TO FAST

-SET CPU CLOCK FREQ TO NORMAL

-SET CONSOLE DEFAULTS

NOTE: <OPTIONS> ARE:

OCTAL, HEX, PHYSICAL, VIRTUAL, INTERNAL

GENERAL, VBUS, IDBUS, BYTE, WORD, LONG, QUAD

-PUT <NUMBER> INTO CONSOLE RELOCATION

REGISTER. RELOCATION REGISTER IS

ADDED TO EFFECTIVE ADDRESS OF

PHYSICAL AND VIRTUAL EXAMINES AND

DEPOSITS.

-SET 'STOP ON MICRO-MATCH' ENABLE

-ENABLE SINGLE BUS CYCLE CLOCK MODE

-ENABLES SINGLE INSTRUCTION MODE

-ENABLE SINGLE TIME STATE CLOCK MODE

-SET FILL COUNT FOR # OF BLANKS

WRITTEN TO THE TERMINAL AFTER <CRLF>

-PUT CONSOLE TERMINAL INTO 'PROGRAM

I/O' MODE

-SHOWS CONSOLE AND CPU STATE

-SHOWS VERSIONS OF MICROCODE AND

CONSOLE

-INITIALIZES THE CPU, DEPOSITS <ADDRESS>

TO PC, ISSUES A CONTINUE TO THE ISP.

-RUNS MICRO-DIAGNOSTICS

-LOADS MICRO-DIAGNOSTICS, AWAITS

COMMANDS

-UNJAMS THE SBI

-CALLS MICRO-DEBUGGER. WCS MICRO-

DEBUGGER FLOPPY MUST BE INSERTED

IN CS1. ELSE, "FILE NOT FOUND" ERROR.

(FOR DEBUGGER HELP, INSERT WCS DEBUG

FLOPPY, THEN TYPE 'awcsmon.hlp')

-WHEN EXECUTED FROM AN INDIRECT

COMMAND FILE, THIS COMMAND WILL CAUSE

COMMAND FILE EXECUTION TO STOP UNTIL:

A) A 'DONE' SIGNAL IS RECEIVED FROM

THE PROGRAM RUNNING IN THE VAX

(COMMAND FILE EXECUTION WILL

CONTINUE), OR

B) THE VAX-11/780 HALTS, OR OPER-

```

!
!
!
! ^P (CONTROL-P)
!
!
! a<FILENAME>
!
! <END OF 'CONSOL.HLP'>

```

```

      ATOR TYPES A CONTROL-C (^C :
      COMMAND FILE EXECUTION WILL
      TERMINATE).
-PUT CONSOLE TERMINAL INTO 'CONSOLE
I/O' MODE
  (UNLESS MODE SWITCH IN 'DISABLE')
-PROCESS AN INDIRECT COMMAND FILE

```

ZZ-ESKAA-10.1 Table of contents
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56
Table of contents

1- 59 **** VAX11/780 CONSOLE(RAM) VERSION V10-01-LOCAL ****
3- 114 VERSION HISTORY -- ELIT ARCHIVE
4- 898 CONSOLE ASSEMBLY AND LINK NOTES
5- 951 DECLARATIONS AND MACROS
6- 1259 MACRO DEFINITIONS FOR STAR CONSOLE
9- 1555
9- 1556 CONSOLE FLOPPY BOOT
11- 1765 LOAD CONSOLE PROGRAM
12- 1894
12- 1895 COMMAND GETTER
13- 1967 GET A COMMAND LINE
14- 2136 CONSOLE NULL LOOP
17- 2273
17- 2274 COMMAND EXECUTER
18- 2324 COMMAND EXECUTION RTN REGISTER USAGE SUMMARY
19- 2338 BOOT,PROCESS INDIRECT FILE,CLEAR SOMM,CONTINUE
20- 2438 START,UNJAM
21- 2489 HALT,INITIALIZE
22- 2543 NEXT(PERFORM A STEP)
23- 2600 QUAD CLEAR
24- 2650 SET STEP,CLOCK,SOMM
25- 2724 EXAMINE,DEPOSIT
28- 2887 MICRO-ASSISTED EXAMINE/DEPOSIT ROUTINES
29- 2963 EXAMINE ID BUS
30- 2999 EXAMINE/DEPOSIT STAR PC
31- 3017 VBUS EXAMINE
33- 3089 EXAMINE INSTRUCTION REGISTER(IR)
34- 3137 SHOW CONSOLE STATE
35- 3206 SHOW VERSION INFO
36- 3257 SET DEFAULTS
37- 3284 LOAD MICRO-DIAGNOSTIC MONITOR OR MICRO-DEBUGGER
38- 3326 WAIT FOR DONE,SET/CLR MEMORY MAPPING ENABLE
39- 3364 CLOCK TICK REPORTING
40- 3410 CHECK FOR CLOCK STOP,WAIT FOR MICRO-RESPONSE
41- 3459 TEST FOR A MICRO-ROUTINE ERROR
42- 3513 TEST FOR A STAR CPU HALT, REPORT A HALT
44- 3655 PUSH MICRO-STACK,READ/WRITE ID BUS REGISTERS
45- 3727 TEST FOR STAR CPU RUNNING
46- 3754 TEST FOR A MICRO-MACHINE TIME OUT
47- 3808 PCS,WCS,FPLA VERSION CHECKING
48- 3919 READ ID BUS REGISTER ROUTINE
49- 3939 FILENAME CONVERSION TO RAD50
50- 4046 LOAD A FILE
50- 4194 LINK COMMAND
51- 4203 INDIRECT COMMAND LINE RETRIEVER
52- 4263 OPEN FILE,TYPE FLOPPY ERROR MESSAGE
53- 4316 TIMEOUT/ODD ADDRESS TRAP CATCHER
55- 4382 APT 'X' COMMAND EXECUTION
56- 4470
56- 4471 PARSING TABLES AND ACTIONS
57- 4553
57- 4554 PARSE
58- 4669 REMOVE BLANKS,COMPUTE NEXT NODE ADDRESS
59- 4711 RECOGNIZE A STRING OF ASCII CHARACTERS
60- 4753 CHECK FOR A DELIMITER IN INPUT STRING

ZZ-ESKAA-10.1 Table of contents
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56
Table of contents

61- 4791	RECOGNIZE AND CONVERT A NUMERIC ASCII STRING
63- 4904	MAIN SYNTAX CHECK TREE
65- 5032	QUALIFIER SYNTAX CHECK TREE
66- 5056	MAINTREE AND QUALIFIER TREE LISTS
67- 5096	PARSER ACTION ROUTINES
68- 5109	ACTIONS THAT SAVE OPERATION TO PERFORM
68- 5186	ACTIONS FOR QUALIFIERS AND SET DEFAULT COMMAND
70- 5269	SYMBOLIC REGISTER ADDRESS SETUPS
71- 5296	ACTIONS FOR SYMBOLIC ADDRESSES
72- 5337	REGOGNITION STRINGS
73- 5474	TEXT STRING STORAGE
74- 5616	TEMPORARY STORAGE
84- 8367	
84- 8368	CONSOLE SWITCH POSITION CHECKER
85- 8435	CONSOLE SWITCH MODE CHANGE
88- 8795	EMT DESPATCHER FOR EXTRA EMT CODES.
89- 8833	CONSOLE TEMPORARY STORAGE
90- 8873	IMPURE AREA FOR DRIVERS AND FILESERVICES
91- 8972	DEVICE REQUEST QUEUES
92- 9090	RING BUFFER DESCRIPTOR BLOCKS

```
1          ;VAX 11780 CONSOLE -- M.J. HARE, D. EARLE, D. MONROE, I.A. LOUGHLIN
2
3          .LIST MC
4          .NLIST ME,MD,CND
5
6          ;
7          ; IDENTIFICATION MACROS :
8          ;
12
53
54
55          000001          PVER=1
56          000000          SVER=0
57          000000          PEDT=0
58          000001          SEDT=1
59 000000          IDENT \PVER,\SVER,\PEDT,\SEDT,<VAX11/780 CONSOLE(RAM)>
                    .TITLE V10-01-L
                    .SBTTL **** VAX11/780 CONSOLE(RAM) VERSION V10-01-LOCAL ****
                    .IDENT /V1001/
60
65
```

113
114
115
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541

.SBTTL VERSION HISTORY -- EDIT ARCHIVE

;***** V02

; EDIT-00 25-JUL-78

- ; A) RENAME CONSOLE VERSION LEVEL TO '02' TO REFLECT
; IMPLEMENTATION AS A FUNCTIONAL BASELINE LEVEL RD
; CONSOLE.
; B) MOVE THE 'REMOTE FLOPPY DISABLE' FLAG (ALLOC) TO
; AN AREA THAT IS NOT CONDITIONALLY ASSEMBLED.

; EDIT-01 28-JUL-78

- ; A) DUE TO A DISAGREEMENT IN SPECIFICATIONS, THE
; CONSOLE WAS NOT PUTTING THE PC, PSL, AND HALT
; CODES IN THE REGISTERS THAT VMS EXPECTED AFTER
; AN AUTO-RESTART IS INITIATED. THIS VERSION OF
; THE CONSOLE PUTS THESE PARAMETERS IN THE REGISTERS
; THAT VMS EXPECTS. ALSO FIXED A PROBLEM CAUSING
; THE 'HALT REASON' CODE TO BE CLEARED.

; EDIT-02 10-AUG-78

- ; A) DUPLICATE THE KEYBOARD INTERRUPT SERVICE ROUTINE
; ('KBDDBGN') IN RAM, WITH TWO CHANGES:
; 1 - THE CONSOLE WILL NO LONGER RECOGNIZE A
; CONTROL-C AS A REBOOT, WHEN NO USER REQUEST
; IS ACTIVE.
; 2 - CONTROL-P FROM THE LOCAL TERMINAL WILL NO
; LONGER DISABLE 'TALK' IF THE KEYSWITCH IS
; IN REMOTE POSITION.
; THESE CHANGES ARE IN RESPONSE TO F.S. PROBLEMS.
; THE KEYBOARD INTERRUPT VECTOR 'KBDINT' HAS BEEN
; CHANGED TO POINT TO THE NEW RAM ROUTINE.
; B) CUT 'RECSIZ' AGAIN, TO ALLOW ROOM FOR (A).

; EDIT-03 24-AUG-78

- ; A) NOW CLEAR THE PSW, BEFORE TRYING TO SEND A
; CHARACTER TO VMS, WHILE IN PROTOCL PROGRAM I/O
; MODE ; THUS FORCING AN INTERRUPT. THIS
; CHANGE WAS MADE TO EDIT-01-24-A, WHERE A TIMING
; LOOP WAS IMPLEMENTED (KLUDG0), AS STAR NEVER GOT
; MORE THAN NINE CHARACTERS.

; EDIT-04 2-DEC-78

- ; A) CHANGED THE VECTOR 'PRTINT' TO REFLECT 'KLUDG2',
; AND RELATED CHANGES, SO THAT THE RAM ROUTINE IS NOW
; APPARENT IN THE EDITS.
; B) NOW DISABLE LSI INTERRUPTS (PS=340) WHILE FIDDLING
; WITH CIB INTERRUPT ENABLES, IN 'DOCONT', 'SAWHLT',
; ETC.
; C) RE-DIRECTED THE REMOTE TERMINAL INTERRUPT VECTOR TO
; 'RMTENT' IN RAM, TO REFLECT (AND CORRECT) EDIT-V02-
; 02, 'KLUDG3'. NOW THE CONSOLE WILL NO LONGER RE-
; BOOT ON RECEIPT OF A CONTROL-C FROM THE REMOTE

A) FIXED THE TRANSITION INTO REMOTE MODE; AN EDITING MISTAKE CAUSED LOCAL COPY AND CONTROL TO BE SET, INSTEAD OF CLEARED.

B) USED THE PROTOCOL-BUFFERS-FULL FLAG 'MESFLG' TO IMPLEMENT A DUMMY LOOP TO EXECUTE UNTIL BUFFER

597 :
598 :
599 :
600 :
601 :
602 :
603 :
604 :
605 :
606 :
607 :
608 :
609 :
610 :
611 :
612 :
613 :
614 :
615 :
616 :
617 :
618 :
619 :
620 :
621 :
622 :
623 :
624 :
625 :
626 :
627 :
628 :
629 :
630 :
631 :
632 :
633 :
634 :
635 :
636 :
637 :
638 :
639 :
640 :
641 :
642 :
643 :
644 :
645 :
646 :
652 :
653 :
654 :
655 :
656 :

SPACE IS FREED UP, WHILE NOT (!) IN PROGRAM I/O
MODE IN PROTOCOL.

EDIT-09 22-JAN-79
A) CHANGED THE HANDLING OF APT PROTOCOL OUTPUT BUFFERS
FULL, FOR THE LAST TIME. NOW, IF IN CONSOLE MODE,
WITH BUFFERS FULL, WE ENABLE INTERRUPTS IN 'PUTAPO'
AND SPIN IN AN ENDLESS LOOP, WITHIN THE ROUTINE,
UNTIL ONE OF THE BUFFERS GETS FREED UP ON A POLL.
IF IN PROGRAM I/O MODE, WE WILL INITIALLY STALL ON
SETTING 'TX-READY' BY SETTING 'MESFLG'; IF THE
ROUTINE IS RE-ENTERED ONCE THE FLAG IS SET, WE
WILL SPIN IN A LOOP AS FOR CONSOLE MODE.
THESE CHANGES IMPACT THE CIB 'TX-READY' INTERRUPT
SERVICE ROUTINE, WHICH WAS MOVED
B) FIXED AN ELUSIVE BUG WHERE THE CONSOLE WAS PUTTING
A '13.' IN THE LAST BYTE OF APT PROTOCOL OUTPUT
BUFFER, INSTEAD OF THE CORRECT BYTE OF ASCII TEXT.
THE TYPE-BYTE FOR ASCII TEXT BLOCK WAS BEING OVER-
LAID DUE TO THE 'COUNT-BACKWARDS-ONE' LOGIC.

EDIT-10 13-FEB-79
A) CHANGED THE FORMAT OF CONSOLE LISTING TO BE MORE
CONSISTENT, AND COMBINED THE MACRO DEFINITIONS IN
'STRMAC.MAC' WITH THE CONSOLE MAIN SOURCE FILE.
ADDED SOME COMMENTS, SUBTITLES, AND FORMALIZED AN
'ASSEMBLY AND LINK NOTES' SECTION. PLACED THE
IDENTIFICATION MACROS SO THAT THE LISTING WOULD BE
HEADED PROPERLY. DID LOTS OF OTHER SIMILAR LITTLE
THINGS WHICH BASICALLY CHANGE ONLY THE APPEARANCE
OF THE LISTINGS.

***** V03

EDIT-00 15-FEB-79
A) RENAME CONSOLE TO REFLECT FULL R.D. FUNCTIONALITY.

***** V04

EDIT-00 16-FEB-79
A) RENAME CONSOLE VERSION TO '04' TO AVOID CONFUSION
WITH PX-03 (PROTOTYPE) RELEASES.

EDIT-01 21-MAR-79
A) ADDED THE 'XLOAD' COMMAND TO LOAD BINARY DATA; THIS
COMMAND IS VALID IN THE REMOTE CONSOLE VERSION,
WHEN LOADED BY APT, ONLY.
B) REVISED ABSOLUTE ADDRESS REFERENCES TO RELOCATABLE
RELATIVE ADDRESSES, TO ENABLE ASSEMBLY AND LINKING
UNDER RSX11M.

EDIT-02 8-APR-79
A) RE-WROTE THE WAY 'IDENT' AND 'SPMES' MACROS WORK,
SO THAT VERSION AND EDIT NUMBERS ONLY MUST BE
CHANGED IN ONE PLACE. HOWEVER, THE NUMBERS ARE

657 :
658 :
659 :
660 :
661 :
662 :
663 :
664 :
665 :
666 :
667 :
668 :
669 :
670 :
671 :
672 :
673 :
674 :
675 :
676 :
677 :
678 :
679 :
680 :
681 :
682 :
683 :
684 :
685 :
686 :
687 :
688 :
689 :
690 :
691 :
692 :
693 :
694 :
695 :
696 :
697 :
698 :
699 :
700 :
701 :
702 :
703 :
704 :
705 :
706 :
707 :
708 :
709 :
710 :
711 :

NOW IN OCTAL, LIMITING THE DIGITS TO 0 THROUGH 7.

EDIT-03 11-APR-79
A) NOW INHIBIT ERROR MESSAGES PRINTING DURING AN
'X' COMMAND BINARY LOAD, UNDER APT, FROM DEPOSIT
ROUTINES.

EDIT-04 2-OCT-79
A) REVERSED THE ORDER OF 'COUNT' AND 'ADDRESS' IN THE
'X' COMMAND. THE SPEC SAYS THAT THE ADDRESS IS
FIRST FOLLOWED BY THE COUNT.

EDIT-05 18-DEC-79
A) FIXED BUGS IN 'X' LOAD COMMAND. GETTING BYTE COUNT
OF COMMAND STRING AND COUNTING THE CARRIAGE RETURN
IN THE CHECKSUM.

EDIT-06 9-JAN-80
A) DISABLED ECHO OF COMMAND STRINGS IF LOADED BY
APT MANUFACTURING. THIS IS AN ATTEMPT TO RUN AT
9600 BAUD.

EDIT-07 17-JAN-80
A) 'RMRXDN' ROUTINE MUST SAVE EACH CHARACTER RECEIVED
BECAUSE COMMAND CHECKSUM ON 'X' COMMAND COMES IN
BEFORE THE COMMAND LINE CAN BE PARSED.
B) DELETED 'TSTREM' AND 'TSTDIS' ROUTINES AND CHANGED
CALLS TO CALL THE EQUIVALENT ROUTINES IN ROM.

EDIT-08 18-JAN-80 (EDIT 10 OCTAL)
A) CHANGED 'PUTAPO' ROUTINE TO STALL IF ALTERNATE
BUFFER IS MORE THAN HALF FULL AND MAIN BUFFER
IS STILL BLOCKED. STALL IS
APPROXIMATELY ONE CHARACTER TIME AT 300 BAUD
(33 MILLISECONDS).
B) CHANGED CRC ROUTINE TO CALCULATE THE LOW BYTE
OF THE CRC INSTEAD OF TABLE LOOKUP. THIS SAVES
SOME MEMORY SPACE.
C) CHANGED 'RMTENT' SO THAT A CTRL P IS TURNED INTO
A CTRL C IF LOADED BY APT.

EDIT-09 23-JAN-80 (EDIT 11 OCTAL)
A) CHANGED GET COMMAND LINE INTERRUPT ROUTINE TO
SET THE XLOFLG IF THE FIRST CHARACTER OF THE
COMMAND LINE IS AN 'X'.
B) CHANGED 'RMRXDN' ROUTINE TO SKIP PROTOCOL CHECK
IF THE XLOFLG IS SET.
C) CHANGED THE X COMMAND EXECUTION ROUTINE TO DO THE
MEMORY DEPOSITS FASTER.
D) CHANGED THE SWITCH CHANGE ROUTINE TO PERFORM THE
CORRECT SETUP WHEN ENTERING LOCAL, AND REMOTE
DISABLE POSITIONS.
E) CARRIER ERROR MESSAGE IS ENABLED FOR LOCAL/TALK
MODE AND DISABLED WHEN CTRL P RECEIVED FROM
EITHER TERMINAL IN LOCAL/TALK MODE.

712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766

***** V05

EDIT 00 -- 16 JUN 1980

A) ADDED CODE TO SUPPORT EUROPEAN MODEM CONTROL.
CODE IS CONDITIONALLY ASSEMBLED WITH THE PARAMETERS
'CCITT5' (FOR 50 HERTZ CLOCK) OR 'CCITT6' (FOR
60 HERTZ CLOCK, U.S. DEBUG). NOTE THAT EUROPEAN
VERSION USES 'CCITT5'.

B) FIXED BUG CAUSING SYSBOOT ERROR MESSAGE RELATING
TO WCS/FPLA VERSION MISMATCHES.

EDIT 01 -- 26 SEP 1980

A) FIXED BUG THAT CAUSED 'REBOOT' COMMAND TO TRY
AND LOAD WCS IF 780 IS NOT HALTED.

EDIT 02 -- 5 JAN 1981

A) ADDED TEST AND TYPEOUT FOR G & H FLOATING POINT FPLA.
G&H FLOATING POINT IS DETERMINED TO BE PRESENT BY
LOOKING AT BIT 0 FPLA OUTPUT OF LOCATION 085(X).
IF BIT 0 IS SET, G&H IS PRESENT.

B) ADDED CALL TO ROUTINES THAT GET/CHECK/TYPOUT THE
WCS/PCS/FPLA VERSION NUMBERS AFTER WCS LOAD COMMAND.

C) MODIFIED WCS LOAD FUNCTION TO LOAD A MAXIMUM OF 2K
MICRO WORDS IF G&H FPLA OPTION IS NOT PRESENT.

D) CHANGED THE LENGTH OF THE 'USRBUF' FROM 512 BYTES TO
384 BYTES. THIS GAINS 128 BYTES IN CONSOLE OVERLAY
CODE SECTION.

E) ADDED NEW EMT FUNCTION SO THAT THE KEY SWITCH CAN BE
CHECKED WHEN RUNNING MICRO DIAGNOSTICS. THIS SHOULD
FIX THE PROBLEM OF BEING IN THE WRONG STATE IF A
REMOTE DISCONNECT OCCURS WHILE RUNNING MICRO DIAG'S.

EDIT 03 -- 27 JAN 1981

A) MOVED MICRO CODE OPTION FLAG TO VMS EXAMINE AREA
SO WCS LOADER PROGRAM CAN TELL IF G&H IS PRESENT.

EDIT 04 -- 12 FEB 1981

A) MODIFIED 'GETVER' ROUTINE TO FORCE MICRO MACHINE
BACK TO UPC FF(X) BEFORE FREE RUNNING THE CLOCK.

EDIT 05 -- 25 FEB 1981

A) FIXED BUGS IN 'DOLOAD' ROUTINE THAT LOADED TOO MANY
BYTES. ALSO CHANGED TYPEOUT, IF LOADING WCS, TO
'XXXX MICROWORDS LOADED' INSTEAD OF 'XXXX BYTES
LOADED'.

B) CHANGED 'G & H PRESENT' MESSAGE TO 'KE780 PRESENT'.

C) REDUCED 'USRBUF' TO 256 BYTES.

EDIT 06 -- 6 MARCH 1981

A) ADDED ROUTINE TO 'GETVER' ROUTINE TO GET THE SIZE
OF WCS PRESENT IN THE MACHINE. THIS INFORMATION
IS STORED IN 'WCSSIZ', IN MICRO WORDS.

B) ADDED A TEST, IN WCS LOADER, TO CHECK IF LOAD WAS
LARGER THAN THE WCS SIZE. IF IT WAS, PRINT OUT

767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821

IS LIMITED TO THE WCS SIZE.
C) ADDED ADDRESS LIMIT CHECK TO EX/DE CONSOLE SPACE.
D) FOUND THAT THE SYMBOL 'CCITT' ENABLES ASSEMBLY OF
EUROPEAN CONSOLE NOT 'CCITT5' OR 'CCITT6'. FIXED
CLOCK TIME TABLES TO ASSEMBLE THE CORRECT TIME
FOR 50 HERTZ MACHINES.

EDIT 07 -- 9 MARCH 1981

A) DELETED ROUTINE IN EDIT 6-A AND TEST IN EDIT 6-B.
B) MODIFIED WCS LOADER ROUTINE TO ONLY LOAD 3K OF
OF MICRO CODE IF KE780 OPTION IS PRESENT.

EDIT 10 -- 23 MARCH 1981

A) FORCED START ADDRESS OF "LOAD/WCS" COMMAND TO BE
1000(X) IF "/S" QUALIFIER NOT SPECIFIED.
B) MODIFIED "XLATFN" ROUTINE TO SET "DEFNAM" BIT IN
FLAG. THIS WILL CAUSE A "LOAD/WCS" COMMAND WITH
NO FILENAME TO LOAD THE WCSAXX.PAT FILE.

***** V06

EDIT 00 -- 4 MAY 1981

A) RELEASE FOR TEST.

EDIT 01 -- 19 MAY 1981

A) ADDED 'WCSPRES' BIT TO 'FLAG'. THE BIT IS CLEARED
ANY TIME THE CONSOLE IS OVERLAID. IT IS ONLY SET
BY LOADING WCS. 'TSTVER' ROUTINE WILL TAKE FATAL
MISMATCH EXIT IF BIT IS CLEAR.
B) FIXED BUG IN 'WRTREM' ROUTINE THAT CAUSED
APT/MANUFACTURING CONSOLE TO CRASH.

EDIT 02 -- 6 AUG 1981

A) ADDED CODE TO 'RSTPCL' TO CLEAR DOWN THE LOCAL
AND REMOTE(NON PROTOCOL) WRITE QUEUE'S. THIS
SHOULD FIX PROBLEM OF DISCONNECTS AND RECONNECTS
IN EUROPEAN CONSOLE.

EDIT 03 -- 10 AUG 1981

A) FIXED BUG IN CODE IN EDIT 2 ABOVE. ALSO, THERE IS
NO ROOM TO CLEAR DOWN LOCAL QUEUE. AFTER TESTING,
I DON'T THINK IT'S REQUIRED ANYWAY.
B) FIXED BUG THAT CAUSED THE "/NEXT" QUALIFIER NOT TO
WORK WITH NUMBERS LARGER THAN 7FFF(X). CAUSED BY
A SIGNED BRANCH AFTER DECREMENT IN LOOP CONTROL.

EDIT 04 -- 11 AUG 1981

A) CHANGED THE 'CHKSWH' ROUTINE SO THAT THE AUTO-RESTART
SWITCH IS SENSED INDEPENDENT OF THE KEY SWITCH
POSITION. NOTE, THAT THIS DOES NOT LOCK THE AUTO
RESTART SWITCH WHEN IN A 'DISABLED' POSITION.

EDIT 05 -- 17 AUG 1981

A) 'TIMEIT' CALL IS REALLY ONLY .5 SECONDS, NOT 2
SECONDS. THIS CAUSES REMOTE RECEIVER INTERRUPT
ROUTINE TO DROP DTR BEFORE CARRIER PRESENT HAS
TIME TO ASCERT. THIS BUG ONLY FAILS WITH DF02
MODEM IN U.S. REMOTE CONSOLE. MODIFIED WAIT FOR
CARRIER TO BE 20 SECONDS.

EDIT 06 -- 20 AUG 1981

A) MODIFIED 'RMRXDN' (REMOTE INTERRUPT SERVICE) TO
DO THE RIGHT THING (I THINK) WITH DTR IN THE
U.S. REMOTE VERSION.

822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876

: EDIT 07 -- 27 AUG 1981
: A) MODIFIED 'CHKSWH' ROUTINE TO CLEAR 'LOCAL CONTROL'
: FLAG WHEN ENTERING REMOTE DISABLE.
: B) MODIFIED EXIT PROTOCOL ROUTINE TO REENABLE CIB
: INTERRUPTS UNCONDITIONALLY.
: C) MODIFIED CCIIT MODEM HANDLING ROUTINE TO WAIT
: 4 SECONDS BEFORE REASCERTING DTR AFTER ABORTING
: A CALL.
: ***** V07
: EDIT 00 -- 14 SEP 1981
: A) FIXED BUG INTRODUCED IN VERSION 6.3(B) CAUSING
: 'NEXT' QUALIFIER NOT TO WORK.
: ***** V08 (10 OCTAL)
: EDIT 00 -- 09 MAY 1983
: A) CHANGED CODE TO SUPPORT UPLINE TRANSFER OF BINARY
: FILES OVER THE REMOTE PORT. (DOES A BRANCH EVEN
: WHEN A TAB IS SEEN.)
: B) CHANGED CODE TO SUPPORT DOWNLINE TRANSFER OF BINARY
: DATA OVER THE REMOTE PORT. (CLEARS ENTIRE UPPER BYTE
: OF THE RECEIVED DATA.)
: NOTE: Edits to change to version 8 are designated by '**8**' in the comment
: ***** V09 (11 OCTAL)
: EDIT 01 -- 23 OCT 1985
: PROBLEM:
: EU00257
: ITTERRMITTENT CONSOLE.SYS HALT AT 146760 CAUSED BY
: PROCESSOR INTERRUPTS WHEN WE ENTER A 'PUTWRD' ROUTINE.
: RESOLUTION:
: A) SET PRIORITY TO DISABLE INTERRUPTS WHEN ENTERING THE
: 'PUTWRD' ROUTINE VIA A JSR.
: B) SET TIMEOUT PARAMETER TO A LARGER VALUE TO GIVE
: STAR MORE TIME TO GET A CHARACTER. WE DO NOT WANT TO
: DROP ANY CHARACTERS WHEN TRANSFERRING FROM A REMOTE
: SITE.
: RESOLUTION TO THIS PROBLEM CAME FROM REMOTE SERVICES
: EUROPE.
: PROBLEM:
: EU00169
: ITTERRMITTENT HALT AT ADDRESS 35076 WHEN DOING A FILE
: TRANSFER FROM A VAX SYSTEM TO THE RDC HOST VIA THE
: REMOTE CONSOLE. CAUSE: QBUS ERROR WHEN CONSOLE TRIES
: TO EXECUTE A BIT SET INSTRUCTION TO THE TX RDY BIT.
: THE ERROR OCCURS IN 'TXSETR' THE MOST ACTIVE ROUTE TO
: GET HERE IS VIA THE CIB TXRDY INTERRUPT ROUTINE
: 'TXRENT'. TXRENT JUMPS US TO THE 'SENDST' AND 'PUTWRD'
: ROUTINES WITH INTERRUPTS ENABLED. ALLOWING THE
: POSSIBILITY OF HAVING THE QUEUE AND LSI/VAX PROTOCOL
: CORRUPTED.

877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896

:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:
:

RESOLUTION:

SET PRIORITY TO DISABLE INTERRUPTS AFTER ENTERING THE
'TXRENT'.
CHECK THE RDY BIT AND IF SET DON'T BOTHER IT.

SOME MORE EDITS:

SOME OF THESE EDITS WERE INCORPORATED IN THE 785 CONSOLE
HOWEVER WERE NOT IMPLIMENTED IN THIS 780 CONSOLE.

1) FIXED AN ADDRESSING MODE BUG AT LINE 1703. THE
'BIT #ENDBLK,R2' IS CHANGED TO 'BIT #ENDBLK,(R2)'.
COMMENTS ARE AT THAT LOCATION.

2) ELIMINATED CHARACTER CHECKING ON RD UNLOADS.
TAB-TO-SPACE CONVERSION AND FILL CHARACTERS AFTER <CR>
OR <LF> CAUSE BINARY TRANSFER PROBLEMS.

NOTE: Edits to change to version 9 are designated by '**9**' in the comment

```
898 .SBTTL CONSOLE ASSEMBLY AND LINK NOTES
929 :
930 : CONDITIONAL ASSEMBLY FLAGS :
931 :
932 : 'REMVER' IF DEFINED, REMOTE SUPPORT IS INCLUDED
933 : IF UNDEFINED, NO REMOTE SUPPORT
934 :
935 : 'APTDBG' IF DEFINED, ALL UNEXPECTED TRAPS WILL HALT
936 : AT LOCATION 26 (DEBUGGING AID)
937 : IF UNDEFINED, TRAPS THROUGH NORMAL VECTORS
938 :
939 : 'NOTLSI' IF DEFINED, ASSEMBLY WILL CAUSE 'MOVTOPSW'
940 : MACRO TO ASSUME NO 'MTPS' INSTRUCTION
941 : IS AVAILABLE
942 : IF NOT DEFINED, 'MOVTOPSW' MACRO ASSUMES USE
943 : OF 'MTPS' IS LEGAL
944 :
945 : 'APTBLD' BUILDS A SPECIAL APT VERSION OF REMOTE CONSOLE
946 :
947 : 'CCITT' IF DEFINED, CCITT SUPPORT IS INCLUDED
948 : IF NOT DEFINED, NO CCITT SUPPORT
949 :-
```


.SBTTL DECLARATIONS AND MACROS

;GLOBAL DECLARATIONS (USED EXTERNAL TO CONSOLE PROGRAM)

; 1) NON-RELOCATABLE TEMPS, REFERENCED BY ROM-RESIDENT SOFTWARE

.GLOBL SHIFTS,CNVCNT,RADIX,LENGTH,CONTMP,TEMSTR,KDNVEC
.GLOBL KUSCNT,KBFADD,KBYCNT,RXCQE,SPCCNT,SPCCHR,FLPTIM
.GLOBL TERFIL,POSCNT,RXLQE,CONRES,NEWCOD,NEWEMT,DEADHK
.GLOBL NXTSEG,STRIBL,SECNUM,MESADD,NOBYTS,WRTMP
.GLOBL ECHOSV,STARCR,WAITPT,FILERR,DIRENT,AVAILP
.GLOBL WBFNT,WBNVEC,WBTCNT,RXSTSC,RXSPFC,RXBFAD,RXBTCT
.GLOBL RXDNVC,WRTQUE,USRREQ,KBDON,PRTDON,SPCFLG,ERRCOD
.GLOBL ROFLAG,SAVER,RXERRO,FRQDON,FDRV1,TSTHLP,WAITLK
.GLOBL FLAG,LINGOT,RXFUN2,BYTCNT,BUFAD,RXTRY
.GLOBL INTINT,RXLSN,PHYTRK,EFINST,DEFRAD,PGMIOM,PASS1
.GLOBL TCTFLG,MICFLG,TRBYT,CUTOFF,CKXMT1,CHKLC1,BUF1PT
.GLOBL BOOTFL,TIMFLG,RMXCSR,RMRBUF,RMRCR,CHKFLP
.GLOBL NOREMT,NODRV1,CHKXMT,APTBF0
.GLOBL LTHEBF,RTEHBF,FILLP,EMPTY,QUECNT,FLDTFL,BUFNT
.GLOBL KOUNTR,FLPFCT,DATVEC,FLPSTA,FSECTOR,FTRACK,FLDONE
.GLOBL QUEBGN,QUEEND,RMTQUE
.GLOBL REMONL,LASPOS,SYNC,OPNCHK
.GLOBL RMTXPT,EXTKPT,BASEAD

; 2) NON-REDEFINABLE DEFINITIONS USED BY CONSOLE ROM

.GLOBL COMQAL,TOIDHI,TOIDLO,ROUSPR,FMIDLO,FMIDHI
.GLOBL RXDNE,RXDONE,TXRDY,TAREAD,SFWDON,SOFCON
.GLOBL MCS,TLKMOD,LOCCNT,LOCCOP,REMECH
.GLOBL REMOT,LOCKD,FLPYOF,RCSR,RBUF,XCSR,XBUF

; 3) FIXED ROM ENTRY POINTS

ROMBAS=140000 ;CONSOLE ROM BASE ADDRESS

RESTAR=ROMBAS+00 ;CONSOLE REBOOT ADDRESS
CLKSER=ROMBAS+04 ;CLOCK SERVICE VECTOR

;CTXINT=ROMBAS+06 ;TX RDY INTERRUPT SERVICE VECTOR
CTXINT=TXRENT ;"8"

CRXINT=ROMBAS+12 ;RX DNE INT SERV VECTOR
EMTSER=ROMBAS+16 ;EMT TRAP SERVICE VECTOR
CONVRT=ROMBAS+22 ;ASCII CONVERSION RTN VECTOR
DXPREI=ROMBAS+26 ;FLOPPY INT SERV VECTOR

;PRTINT=ROMBAS+32 ;CONSOLE PRINTER INT SERV VECTOR
PRTINT=KLUDG2 ;"8"
;KBDINT=ROMBAS+36 ;CONSOLE KBD INT SERV VECTOR
KBDINT=KLUDG3 ;"8"

951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005

140000

140000
140004

032560'

140012
140016
140022
140026

032300'

032356'

```

1006      140042      LODMIC=ROMBAS+42      ;OVERLAY LOADER ENTRY POINT
1007      140046      TYPEIT=ROMBAS+46      ;MESSAGE TYPING RTN ENTRY
1008      140052      ZFILLP=ROMBAS+52      ;FLOPPY DRIVER EMPTY/FILL BUFFER RETURN
1009      140054      RSAVEP=RCMBAS+54      ;REGISTER SAVING ROUTINE POINTER
1013      140056      OTHRTP=ROMBAS+56      ;ALL OTHER TRAPS VECTOR
1015
1016      ;*****
1017      ;REMENP=ROMBAS+62      ;REMOTE INPUT ENTRY TO KBD SERVICE
1018      036622      REMENP=RMTVEC      ;*
1019      ;*****
1020
1021      140064      GETRNP=ROMBAS+64      ;GET A BYTE FROM RING BUFFER
1022      140066      PUTRNP=ROMBAS+66      ;PUT A BYTE IN A RING BUFFER
1023      140070      PUTAVP=ROMBAS+70      ;RETURN A NODE TO AVAILABLE NODE LIST
1024      140072      WRTLCP=ROMBAS+72      ;WRITE TO LOCAL TERMINAL ONLY ENTRY
1025      140074      RVSTER=ROMBAS+74      ;ENTRY TO REVERSE TERMINAL ADDRESSES
1026      140100      REBCON=ROMBAS+100      ;ENTRY TO REBOOT CONSOLE
1027
1028      ;REGISTER DEFINITIONS
1039
1040      ;FLOPPY AND TERMINAL ERROR CODES
1075
1076      ;CIB DEFINITIONS
1078
1079
1080      173006      IDDATL=173006
1081      173010      IDDATAH=173010
1082      173014      RXDONE=173014
1083      173016      TXREAD=173016
1084      173020      TOIDLO=173020
1085      173022      TOIDHI=173022
1086      173024      FMIDLO=173024
1087      173026      FMIDHI=173026
1088      173030      IDCNTL=173030
1089      173032      MCR=173032
1090      173034      MCS=173034
1091      173036      VBUSR=173036
1092
1093
1094      ;IDCNTL BITS
1095      100000      IDCYCL=100000
1096      000100      IDWRIT=100
1097      000200      IDMANT=200
1098
1099      ;MCR BITS
1100      100000      HLTREQ=100000
1101      010000      CPURES=10000
1102      002000      MAINTR=2000
1103      000400      STRIND=400
1104      000200      ROMNOP=200
1105      000100      SOMMB=100
1106      000040      CLKSTD=40
1107      000010      FREQ0=10

```

```

1108      000020      FREQ1=20
1109      000004      STS=4
1110      000002      SBC=2
1111      000001      PROCED=1
1112
1113      ;MCS BITS
1114      010000      FLPYOF=10000
1115      004000      BOOTBT=4000
1116      000400      RUNBIT=400
1117      ;CNSLAK=200
1118      000100      RDYIE=100
1119      000040      DNEIE=40
1120      000004      AUTORS=4
1121      000002      REMOT=2
1122      000001      LOCKD=1
1123
1124      ;VBUSR DEFINITIONS
1125      ;CPT0=200
1126      ;CPT1=100
1127      ;CPT2=40
1128      000020      CPT3=20
1129      ;SLFTST=4
1130      000002      VLOAD=2
1131      000001      VCLK=1
1132
1133      ;RXDONE BITS
1134      000200      RXDNE=200
1135
1136      ;TXREAD BITS
1137      000200      TXRDY=200
1138
1139
1140      ;STAR CONTROL STORE ROUTINE ADDRESSES
1141      000440      CPHYSE=440
1142      000442      CGREGE=442
1143      000444      CPREGE=444
1144      000447      CONCON=447
1145      000452      SBIUNJ=452
1146
1147      ;ID BUS ADDRESSES
1148      000014      CESREG=14      ;ADDRESS OF 'CES' REGISTER
1149      000026      ID16=26      ;ADDRESS OF ACCELARATOR PC
1150      000040      IDAUST=40      ;ADDRESS OF MICRO-STACK
1151      000042      WCSADD=42      ;ADDRESS OF WCS ADDRESS REG
1152      000043      WCSDAT=43      ;ADDRESS OF WCS DATA REG
1153      000061      T1=61      ;ID BUS TEMP 1
1154      000062      T2=62      ;ID BUS TEMP 2
1155      000063      T3=63      ;ID BUS TEMP 3
1156      000022      TBUF0=22      ;TBUF DATA GROUP 0
1157      000023      TBUF1=23
1158      000031      SBIERR=31
1159      000032      SBIADD=32
1160      000036      CACPAR=36
1161      000056      DSV=56
1162

```

```

1163      ;INTERNAL (PROCESSOR) REGISTER DEFINITIONS
1164      000066      INTR36=66      ;'QUAD CLEAR' REGISTER (HEX ADDRESS 36)
1165
1166      ;'D.SV' ERROR CODES
1167      ;SUCCE=0      ;SUCCESSFUL COMPLETION
1168      000001      MEMFAL=1      ;MEMORY FAULT
1169      000002      CONERR=2      ;ERROR ON CONSOLE REQUEST
1170      ;INITDN=3      ;INITIALIZATION DONE
1171      ;INTINV=4      ;INT STACK NOT VALID
1172      ;DBLHLT=5      ;CPU DOUBLE ERROR HALT
1173      000006      HLTINS=6      ;HALT INSTRUCTION EXECUTED
1174      ;ILLVEC=7      ;ILLEGAL I/E VECTOR
1175      ;NOUWCS=10      ;NO USER WCS
1176      ;INTPEN=11      ;INTERRUPT PENDING ON HALT
1177      ;CHMERR=12      ;CHANGE MODE ERROR
1178      ;PRGERR=13      ;ERROR ON PROCESSOR REGISTER REFERENCE FROM CONSOLE
1179      000013      LASERR=13      ;LAST VALID ERROR CODE *****
1180
1181      000421      PCVERS=421      ;LOC 111(HEX) CONTAINS PCS VERSION
1182      010421      WCVERS=10421      ;LOC 1111(HEX) CONTAINS WCS PRIMARY VESION
1183      007600      FPVERS=7600      ;LOC F80(HEX) CONTAINS FPLA VERSION
1184      000205      MOPTFL=205      ;LOC 85(HEX) CONTAINS MICRO CODE OPTION FLAG
1185      010000      FIRSTW=10000      ;FIRST WCS ADDRESS FOR WCS ECO FILE LOAD
1186      030000      RESLSB=30000      ;WARM RESTART ADDRESS(LSB'S)
1187      020000      RESMSB=20000      ;WARM RESTART ADDRESS(MSB'S)
1188
1189
1190      ;*****
1191      ;NOTE: THESE ADDRESS ASSIGNMENTS MUST NEVER BE CHANGED BECAUSE
1192      ;       THESE POINTERS ARE USED IN ROUTINES BLASTED INTO THE
1193      ;       CONSOLE ROM
1194      ;*****
1195      037766      RMRCSR=37766
1196      037770      RMRBUF=RMRCSR+2
1197      037772      RMXCSR=RMRCSR+4
1198      037774      RMXBUF=RMRCSR+6
1199
1200      000314      RMTXVC=314      ;REMOTE TRANSMITTER INTERRUPT VECTOR
1201      000310      RMRXVC=310      ;REMOTE RECEIVER INTERRUPT VECTOR
1202
1203      100000      DSTINT=100000      ;DATASET INTERRUPT
1204      040000      RINGDT=40000      ;RING DEFECT BIT
1205      020000      CLRSND=20000      ;CLEAR TO SEND
1206      010000      CARDET=10000      ;CARRIER PRESENT
1207      004000      RCVACT=4000      ;RECEIVER ACTIVE
1208      002000      DSTRDY=2000      ;DATASET READY
1209      000200      RCVDON=200      ;RECEIVER DONE
1210      000100      RCVINT=100      ;RCV INT ENA
1211      000100      XMTINT=100      ;XMIT INT ENA
1212      000040      DATINT=40      ;DATA STATUS INT ENA
1213      000004      REQSND=4      ;REQUEST TO SEND
1214      000002      DATRDY=2      ;DATA TERMINAL READY
1215
1216      ;LOCAL TERMINAL ADDRESSES
1217      037756      RCSR=37756
  
```

ZZ-ESKAA-10.1 DECLARATIONS AND MACROS
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 5-4
 DECLARATIONS AND MACROS

```

1218      037760      RBUF=RCSR+2
1219      037762      XCSR=RCSR+4
1220      037764      XBUF=RCSR+6
1221      000064      LCTXVC=64      ;LOCAL TRANSMITTER INTERRUPT VECTOR
1222      000060      LCRXVC=60      ;LOCAL RECEIVER INTERRUPT VECTOR
1223
1224
1225      ;AREA OF CONSOLE MEMORY EXAMINABLE BY STAR
1226      037600      FRFSFIX=37600      ;BASE ADDRESS ADDED TO OFFSET
1227      037744      MICOPT=FRFSFIX+144      ; MICRO CODE OPTION FLAG. BIT<0> CONTAINS THE MICRO
1228                                     ; CODE OPTION OF THE MACHINE. THE REST OF THE BITS
1229                                     ; MUST BE ZERO.
1230      000001      OPTMSK=001      ; MASK FOR MICRO CODE OPTION BITS
1231      000000      NOOPT =000      ; NO OPTION PRESENT
1232      0C0001      GHOPT =001      ; G & H FLOATING POINT PRESENT
1233
1234      ;EDIT-16 IMPLEMENT COLD/WARM START FLAGS
1235      037745      WRMSTR=FRFSFIX+145      ;WARM-START FLAG
1236      037746      CLDSTR=FRFSFIX+146      ;COLD-START FLAG
1237      ;END EDIT-16
1238      037747      APTLOD=FRFSFIX+147      ;NON-ZERO WHEN CONSOLE LOADED BY APT
1239      037750      LASPOS=FRFSFIX+150      ;HOLDS LAST INFO ON MODE BITS FROM MCS(LOWER 2 BITS)
1240      037751      AUTFLG=FRFSFIX+151      ;0 WHEN AUTO-RESTART DISABLED, -1 WHEN ENABLED
1241      037752      PCSVER=FRFSFIX+152      ;PCS VERSION BYTE
1242      037753      WPMVER=FRFSFIX+153      ;WCS PRIMARY VERSION BYTE
1243      037754      WSCVER=FRFSFIX+154      ;WCS SECONDARY VERSION BYTE
1244      037755      FPLVER=FRFSFIX+155      ;FPLA VERSION BYTE
1245      ;
1246      ; LSI-11 PHYSICAL MEMORY LIMIT
1247      ;
1251      040000      MEMSIZ = 40000
1253
1254      ;*****
1255      000000      $REGDF
1256      000000      $CODDF
1257      ;*****

```

```
1259 .SBTTL MACRO DEFINITIONS FOR STAR CONSOLE
1260
1261 ;MACRO DEFINITIONS FOR STAR CONSOLE
1262 ;M.J. HARE -- DECEMBER 1977
1263
1264 ;*****
1265 ; EMT SERVICE MACROS
1266 ;*****
1267
1268 ;INITIALIZE THE TERMINAL (TINIT=EMT 0)
1269
1273 ;WRITE TO THE TERMINAL (TWRITE=EMT 1 ; RMWRON=EMT 15 ; LCWRON=EMT 16)
1274
1304 ;READ FROM THE TERMINAL (TREAD=EMT 2)
1305
1321
1322 ;OPEN A FILE ON FLOPPY DRIVE 0 (OPENFL=EMT 3)
1323
1332
1333 ;READ FLOPPY SECTOR(S) (READSC=EMT 4)
1334
1364 ;WRITE FLOPPY SECTOR(S) (WRITSC=EMT 5)
1365
1366 ;LOAD CONSOLE W/ WCS ECO'S (LOADCN=EMT 6)
1396
1397
1398 ;ASCII OPUT CONVERSION (CNVERT=EMT 7)
1402
1403
1404 ;RETURN DEFAULT RADIX IN R2 (RADGET=EMT 10)
1408
1409
1410 ;OPEN A FILE ON FLOPPY DRIVE 1 (OPNFL1=EMT 11)
1414
1415
1416
1425
1426
1427 ;+
1428 ; TYPEMES MACRO (TYP1=EMT 12 ; TYP2=EMT 13)
1429 ;
1430 ; FORMAT: TYPEMES ARG,,CRFLAG
1431 ;
1432 ; ARG=SOURCE OF STRING TO TYPE, FIRST BYTE IS # OF BYTES IN STRING
1433 ; (IF BLANK, MESSAGE POINTER IS ON STACK)
1434 ;
1435 ; CRFLAG= IF NOT BLANK, TYPE A CR AND LF BEFORE STRING
1436 ; -
1447
1448
1449 ;LOAD CONSOLE W/O WCS ECO'S (LCANWC=EMT 14)
1450 ; .MACRO LDCNNW ;THIS MACRO IS NEVER USED. EITHER RESTART
1451 ; EMT LCANWC ;CONSOL.SYS ALREADY LOADED, OR ELSE DO A
1452 ; .ENDM LDCNNW ;COMPLETE RELOAD WITH WCS. (LOADCN).
1453
```

```
1454
1455      ;                               (RMWRON=EMT 15  -- SEE 'T$WRITE)
1456      ;                               (LCWRON=EMT 16  -- SEE 'T$WRITE)
1457
1458
1459      ;TIME-OUT MACRO                               (TMERTR=EMT 17)
1460      ;      COUNT A DELAY OF APPROX. ONE HALF SECOND
1461
1462
1463
1464
1465      ;RESET LSI-11                               (R$SET=EMT 20)
1466
1467
1468      ;LOAD CONSOL.SYS AND WCS ECO'S. (LDCONS=EMT 21)
1469
1470
1471
1472      ;INDICATE IF CCITT MODEM HANDLING IN USE.      (MDMTYP=EMT 22)
1473
1474
1475      ;CHECK POSITION OF KEY SWITCH                  (CHKSWITCH=EMT 23)
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
```

```
1487 ;*****
1488 ;
1489 ;          UTILITY MACROS
1490 ;*****
1491
1492 ;+
1493 ;MOVE TO PSW MACRO
1494 ;IF 'NEWPSW' FIELD OF CALL IS BLANK, NEW PSW ASSUMED ON STACK
1495 ;-
1512
1513
1514 ;OPEN A FILE
1519
1520
1521 ;CREATE AN ASCII MESSAGE
1528
1529
1530 ;TYPE A MESSAGE (SET-UP FOR TYPMES MACRO)
1536
1537
1538 ;CREATE A ZERO-DATA BLOCK
1542
1543
1544 ;CREATE A BLANK DATA BLOCK STARTING AT PC LOCATION
1545 ; AND IF TO ADDRESS 'NUMB'
1552
```



```
1555 .SBTTL
1556 .SBTTL  CONSOLE FLOPPY BOOT
1557
1558 000000'  BASE=.  ;USED FOR LOCATION COUNTER SETTING
1559
1560 000000 000240 240  ;A NOP FOR LOAD CHECK
1561 000002 000411 BR RETRY
1562
1563 000004 000636' .WORD NOREMO
1564 000006 000340 .WORD 340
1565
1566 002000 PERM=2000
1567 004000 ENDBLK=4000
1568
1569 ;FLOPPY FUNCTION DEFINITIONS
1570 000001 CSGO=1 ;FLOPPY START
1571 000002 CSEBUF=2 ;EMPTY BUFFER
1572 000006 CSRD=6 ;READ SECTOR
1573 000040 CSDONE=40 ;RX DONE
1574
1575 177170 RXCS=177170 ;RXCS STATUS REG
1576
1577
1578 000010 FILLTO 14
1579 000014 000104' .WORD READS ;BPT GOES TO READ SECTOR
1580 000016 000340 .WORD 340 ;PSW GETS 'INTERRUPTS OFF'
1581
1582 000020 000166' .WORD WAITRT ;IOT GOES TO FLOPPY WAIT
1583 000022 000340 .WORD 340
1584
1585 000024 001004' .WORD APTSRT ;APT LOAD STARTUP ADDRESS POINTER
1586 000026 011706 RETRY: MOV @PC,SP ;POINT SP TO "SAFE" ADDRESS (VALUE OF NEXT INSTRUCTION)
1587 000030 012702 000200 MOV #200,R2 ;R2 POINTS TO LOAD ADDRESS OF NEXT PART
1588 ;OF BOOT
1589 000034 005000 CLR R0
1590 000036 005200 INC R0
1591 000040 011703 MOV @PC,R3 ;READ SECTOR 3 NEXT (BPT INSTR. = 003)
1592 000042 000003 BPT ;CALL READ SECTOR
1593 000044 105067 037747' CLR BPT ;REMEMBER CONSOLE BOOTED
1594 000050 000453 BR BOOT2
1595
1596 000052 FILLTO 100
1597
1598 000100 000174' .WORD RTIRET
1599 000102 000340 .WORD 340 ;IGNORE CLOCK INTS WHILE BOOTING
1600 000104 READS: ;READ SECTOR SUBROUTINE
1601 000104 162701 000200 SUB #128,R1 ;R1 HAS # OF BYTES TO BE READ
1602 000110 010146 MOV R1,-(SP) ;SAVE REMAINING # OF BYTES
1603 000112 012701 000200 MOV #128,R1 ;NUMBER OF BYTES IN SECTOR
1604 000116 012704 177170 MOV #RXCS,R4 ;POINT R4 TO FLOPPY STATUS REGISTER
1605 000122 010405 MOV R4,R5 ;R5 ALSO
1606 000124 012725 MOV (PC)+,(R5)+ ;START READ AND POINT R5 TO RXDB
1607 000126 000007 .WORD CSGO+CSRD
1608 000130 000004 IOT ;CALL WAIT
1609 000132 010315 MOV R3,@R5 ;LOAD SECTOR NUMBER
```

```

1610 000134 000004      IOT          ;WAIT
1611 000136 010015      MOV      R0,aR5      ;LOAD TRACK #
1612 000140 000004      IOT          ;WAIT
1613 000142 012714 000003      MOV      #CSGO+CSEBUF,aR4 ;FUNCTION=EMPTY BUFFER
1614 000146 000004      IOT          ;WAIT
1615 000150 105714      4$: TSTB      aR4      ;TEST FOR TR FLAG
1616 000152 100376      BPL      4$          ;BR IF TR NOT UP
1617 000154 111522      MOV      aR5,(R2)+    ;STORE BYTE IN MEM
1618 000156 005301      DEC      R1          ;BYTE CNT MINUS 1
1619 000160 003373      BGT      4$          ;BR IF MORE
1620 000162 012601      MOV      (SP)+,R1    ;RESTORE REMAINING BYTE COUNT
1621 000164 000002      RTI
1622
1623 000166      WAITRT: ;WAIT FOR TR, ERROR, OR DONE
1624 000166 005714      TST      aR4          ;CHECK TR, ERROR, DONE
1625 000170 001776      BEQ      WAITRT      ;BR UNTIL ONE COMES UP
1626 000172 100715      BMI      RETRY      ;START AGAIN IF ERROR (MSB SET)
1627 000174 000002      RTIRET: RTI
1628
1629 000176      FILLTO 200
1630
1631 000200 122323      BOOT2: CMPB      (R3)+,(R3)+ ;SECTOR # TO 5
1632 000202 000003      BPT          ;CALL READ SECTOR
1633 000204 122323      CMPB      (R3)+,(R3)+ ;SECTOR # TO 7
1634 000206 000003      BPT          ;CALL READ SECTOR
1635 000210 012737 000334' 000020      MOV      #TRWAIT,a#20 ;SET NEW TR WAIT VECTOR
1636 000216 000501      BR      BOOT3
  
```

```

1638
1639      : FLOPPY INTERLEAVE ALGORITHM
1640 000220 006300      READ: ASL    R0      ;CHANGE LOGICAL BLOCK # TO LOGICAL SEC #
1641 000222 006300      ASL    R0
1642 000224 006301      ASL    R1      ;WORD COUNT TO BYTE COUNT
1643 000226 010046      10$: MOV    R0,-(SP) ;SAVE SEC #
1644 000230 010003      MOV    R0,R3      ;R3 GETS LOG SEC #
1645 000232 012700 000010      MOV    #8.,R0 ;DIVIDE ROUTINE, R0 IS LOOP COUNTER
1646 000236 022703 006400      1$: CMP    #6400,R3 ;DOES 26 GO INTO DIVIDEND?
1647 000242 101002      BHI    2$      ;BR IF NOT, C BIT CLEAR
1648 000244 062703 171400      ADD    #171400,R3 ;SUBTRACT 26 FROM DIVIDEND
1649 000250 006103      2$: ROL    R3      ;SHIFT DIVIDEND AND QUOTIENT
1650 000252 005300      DEC    R0
1651 000254 003370      BGT    1$      ;MORE TO DIVIDE?
1652      ;END OF DIVIDE : R3 CONTAINS TRACK # IN HIGH BYTE,
1653      ; SECTOR IN LOW BYTE
1654 000256 110300      MOVB   R3,R0      ;R0 GETS TRACK #
1655 000260 105003      CLRB   R3      ;REMOVE TRACK # FROM REMAINDER
1656 000262 000303      SWAB   R3      ;GET REMAINDER (SECTOR #)
1657 000264 022703 000014      CMP    #12.,R3 ;SET C ONLY IF 12 < R3 < 24
1658 000270 005103      ROL    R3      ;DOUBLE SECTOR # FOR 2-TO-1 INTERLEAVE
1659      ;MOVE C-BIT TO LSB FOR SECTOR GROUP
1660 000272 006300      ASL    R0
1661 000274 060003      ADD    R0,R3      ;SKEW BY 6*TRACK # FOR
1662 000276 060003      ADD    R0,R3      ; TRACK ACCESS TIME
1663 000300 060003      ADD    R0,R3
1664 000302 006200      ASR    R0      ;RESTORE TRACK #
1665 000304 005200      INC    R0      ;TRACK # TO RANGE 1-TO-76
1666 000306 162703 000032      3$: SUB    #26.,R3
1667 000312 002375      BGE    3$      ;BR UNTIL SECTOR # IS NEGATIVE
1668 000314 062703 000033      ADD    #27.,R3 ;SECTOR TO RANGE 1 TO 26.
1669 000320 000003      BPT     ;READ SECTOR -- DEC BYTE COUNT BY 128.
1670 000322 012600      MOV    (SP)+,R0 ;R0 GETS LOG SEC # AGAIN
1671 000324 005200      INC    R0      ;BUMP LSN
1672 000326 005701      TST    R1      ;TEST BYTE COUNT
1673 000330 003336      BGT    10$      ;BR IF MORE BYTES TO GET
1674 000332 000207      RTS    PC
1675
1676 000334 005714      TRWAIT: TST    @R4      ;TEST FOR TR, ERROR, DONE
1677 000336 001776      BEQ    TRWAIT ;FUNCTION COMPLETE?
1678 000340 100315      BPL    RTIRET ;NO ERROR IF MSB CLEAR (RTI)
1679 000342 004067 000036      JSR    R0,REPORT
1680 000346 012 077 102      .ASCIZ <12>\?B-I/O ERROR\ ;ERROR MESSAGE TO REPORT
      000351 055 111 057
      000354 117 040 105
      000357 122 122 117
      000362 122 000
1681 000364      FILLTO 400
1682
1683      ;ERROR PRINTER
1684 000400 112037 177566      REPORT1: MOVB   (R0)+,@#177566 ;PRINT A CHARACTER OF ERROR MESSAGE
1685 000404 105737 177564      REPORT: TSTB   @#177564 ;WAIT FOR PRINTER READY
1686 000410 100375      BPL    REPORT
1687 000412 105710      TSTB   @R0      ;TEST END OF MESSAGE
1688 000414 001371      BNE    REPORT1 ;BR IF MORE TO PRINT

```

```

1689 000416 000167 140000' JMP RESTAR ;TRY REBOOTING
1690
1691 ;REMAINDER OF BOOT
1692 000422 012706 037400 BOOT3: MOV #37400,SP ;SET STACK PNTR AWAY FROM BOOT CODE FOR NOW.
1693 000426 013746 037776 MOV a#37776,-(SP) ;STACK 'POWER-UP/CRASH' FLAG
1694 ;(CONTAINS 123456 IF CRASH RECOVERY)
1695 000432 012700 000001 MOV #1,R0 ;GET DIRECTORY SEGMENT
1696 000436 006300 DFND: ASL R0 ;(FIRST DIRECTORY SEGMENT IS SECT 6)
1697 000440 062700 ADD #4,R0
1698 000444 012701 001000 MOV #1000,R1 ;DIR SEG IS 512. WORDS
1699 000450 012702 001000 MOV #BUFFB,R2 ;R2=BUF ADD
1700 000454 004767 177540 JSR PC,READ ;READ SEC
1701 000460 012701 001010 MOV #BUFFB-10,R1 ;R1=STARTING BLOCK WD
1702 000464 012100 MOV (R1)+,R0
1703 000466 010102 MONF: MOV R1,R2 ;R2 GETS ADD OF STAT WD
1704 000470 032721 002000 BIT #PERM,(R1)+ ;TEST FOR PERMANENT FILE
1705 000474 001411 BEQ 1$ ;BR IF NOT PERMANENT
1706 000476 162721 SUB (PC)+,(R1)+ ;LOOK FOR CONSOL.SYS
1707 000500 012446 .RAD50 /CON/
1708 000502 162721 SUB (PC)+,(R1)+
1709 000504 074444 .RAD50 /SQL/
1710 000506 162721 SUB (PC)+,(R1)+
1711 000510 075273 .RAD50 /SYS/
1712 000512 001002 BNE 1$ ;BRANCH IF NOT .SYS EXTENSION
1713 000514 054141 BIS -(R1),-(R1) ;TST BOTH PARTS OF FILE NAME MATCHING
1714 000516 001432 BEQ CONFND ;BRANCH IF CONSOL FOUND
1715 ; MODIFY:
1716 ;1$: BIT #ENDBLK,R2 ;TEST FOR END OF SEGMENT **9**
1717 000520 032712 004000 1$: BIT #ENDBLK,(R2) ;BUG FIX R2 CONTAINS THE ADDRESS OF THE STATUS
1718 ;WORD, NOT THE STATUS WORD ITSELF. THEREFORE
1719 ;MODE 1 SHOULD BE USED RATHER THAN MODE 0.
1720 ;END MODIFY
1721 000524 001010 BNE 2$ ;BR IF END OF SEG
1722 000526 066200 000010 ADD 10(R2),R0 ;INCREASE STARTING BLOCK ADDRESS
1723 000532 062702 000016 ADD #16,R2 ;POINT R2 TO NEXT ENTRY
1724 000536 066702 001006' ADD BUFFB+6,R2 ;ADD IN # OF EXTRA WDS
1725 000542 010201 MOV R2,R1 ;POINT R1 TO NEXT
1726 000544 000750 BR MONF
1727 000546 016700 001002' 2$: MOV BUFFB+2,R0 ;SEE IF NEXT DIR SEG EXISTS
1728 000552 001331 BNE DFND ;BR IF IT EXISTS
1729 000554 004067 177624 JSR R0,REPORT ;REPORT FAIL TO FIND CONSOL
1730 000560 015 012 077 .ASCIIZ <15><12>\?B-NO CONSOL.SYS\<12>
000563 102 055 116
000566 117 040 103
000571 117 116 123
000574 117 114 056
000577 123 131 123
000602 012 000
1731 .EVEN

```

```

1765          .SBTTL    LOAD CONSOLE PROGRAM
1766          :INPUTS:  R0 IS STARTING BLOCK OF CONSOL.SYS
1767
1768
1769 000604 016201 000010      CONFND: MOV     10(R2),R1      ;R1 GETS CONSOL SIZE
1770 000610 124120              CMPB     -(R1),(R0)+      ;ADD 1 TO START BLOCK, SUB 1 FROM # BLOCKS
1771                          ;MULTIPLY R1 BY 256 TO GET # OF WORDS IN CONSOL.SYS
1772 000612 000301              SWAB     R1
1773 000614 012702 001000      MOV     #1000,R2          ;R2 GETS CONSOL BASE ADDRESS
1774 000620 004767 177374      JSR      PC,READ          ;LOAD IN CONSOLE
1775
1776
1777
1791 000624 010046      CONSTR: MOV     R0,-(SP)          ;PUT 2 ON STACK FOR CODE BELOW
1792 000626 010046      MOV     R0,-(SP)
1793 000630 000240      NOP
1794 000632 000240      NOP
1795 000634 000240      NOP
1796
1800 000636 022626      NOREMO: CMP     (SP)+,(SP)+      ;GET PC AND PSW OFF STACK
1801 000640 105067 035326      CLRB     NOREMT          ;NOTE NO REMOTE TERMINAL
1802 000644 012704 177170      5$:     MOV     #RXCS,R4      ;POINT R4 TO FLOPPY CONTROL AND STATUS REG
1803 000650 012714 000033      MOV     #33,(R4)          ;READ DRIVE 1 STATUS
1804 000654 032714 000040      8$:     BIT     #CSDONE,(R4)  ;FUNCTION COMPLETE?
1805 000660 001775      BEQ     8$                        ;BR IF NOT
1806 000662 105737 177172      TSTB     @#RXCS+2          ;DRIVE 1 READY?
1807 000666 100402      BMI     9$                        ;BR IF IT IS
1808 000670 105267 035277      INCB     NODRV1          ;REMEMBER THERE IS NO DRIVE 1
1809 000674 012704 173032      9$:     MOV     #MCR,R4      ;POINT R4 TO MCR REGISTER FOR INIT RTN USE
1810 000700 021627 123456      CMP     (SP),#123456      ;POWER-UP CAUSE OF THIS BOOT?
1811 000704 001004      BNE     12$                      ;BR IF POWER UP
1812 000706 042767 000020 034464      BIC     #INITLD,TCONTL ;PREVENT AUTO-RESTART ON CRASH RECOVERY
1813 000714 000402      BR      11$
1814
1815 000716 004767 002710      12$:     JSR     PC,INITQU      ;INIT STAR CPU & STARLET INPUT QUEUE(EDIT-21A)
1816 000722 042767 000002 034672      11$:     BIC     #SAWHLT,FLAG ;CLEAR 'SAW HALT' BIT OF 'FLAG'
1817 000730 005037 037776      CLR     @#37776          ;CHANGE 'POWER-UP' FLAG TO CRASH RECOVERY VALUE
1818
1819          .DSABL    LSB
1820
1821 000734 012704 000001      MOV     #1,R4              ;CAUSE AN ECO FILE LOAD (IGNORED WHEN APTLOD=1)
1822 000740 000507              BR      CONBOT          ;START UP CONSOLE
1823
1824          DUFFB=1000 ;DIRECTORY BUFFER
1825          P00FSZ=< -BASE+777>/1000
1826 000742          FILLTO 1000
1827
1828 001000 000463      BR      CONSRT          ;USED FOR CONSOLE RELOAD ENTRY
1829          ;*****
1830 001002 102777      BVS          ;USED TO INDICATE CONSOLE PROGRAM LOADED
1831          ;THIS INSTRUCTION MUST APPER AT ADDRESS 1002
1832          ;*****
1833
1834 001004      APTSRT: ;SPECIAL APT START-UP ENTRY
1835          ;REVERSE TERMINAL ADDRESS ASSIGNMENTS, DISABLE FLOPPY USAGE
1836          ;CAUSE ECO LOAD AND SHOW COMMAND TO BE SKIPPED

```

```

1837 001004 112767 000001 037747'      MOVB    #1,APTL0D      ;THIS WILL PREVENT SHOW AND ECO LOAD
1838 001012 012706 001000                MOV     #1000,SP      ;SET STACK
1839 001016 105267 035151                INCB    MCDRV1      ;DISABLE FLOPPY DRIVE 1
1840 001022 105267 035146                INCB    ALLREM      ;FORCE ALL FLOPPY REQUESTS TO APT
1841 001026 005067 037752'                CLR     PCSVER      ;CLEAR VERSION TEMPS TO PREVENT ERRORS CAUSED
1842 001032 005067 037754'                CLR     WCSVER      ;BY LACK OF WCS LOAD
1843 001036 012702 037756                MOV     #RCSR,R2      ;SET TERMINAL ADDRESS ASSIGNMENTS
1844 001042 012700 175610                MOV     #175610,R0    ;THIS ADDRESS WILL BE LOCAL TERMINAL (APT ONLY)
1845 001046 012703 000004                MOV     #4,R3        ;USED TO COUNT 4 ADDRESSES PER TERMINAL
1846 001052 010304                        MOV     R3,R4
1847 001054 010022                        10$: MOV    R0,(R2)+      ;SAVE AN ADDRESS FOR LOCAL TERMINAL (4 WORDS)
1848 001056 005200                        INC     R0
1849 001060 005200                        INC     R0
1850 001062 005303                        DEC     R3
1851 001064 003373                        BGT     10$
1852 001066 012700 177560                MOV     #177560,R0
1853 001072 010022                        20$: MOV    R0,(R2)+      ;SAVE AN ADDRESS FOR REMOTE TERMINAL (4 WORDS)
1854 001074 005200                        INC     R0
1855 001076 005200                        INC     R0
1856 001100 005304                        DEC     R4
1857 001102 003373                        BGT     20$
1858 001104 016746 022404                MOV     BUF0+RMTXVC,-(SP) ;CODE HERE SWAPS INTERRUPT VECTOR CONTENTS
1859 001110 016767 022150 022376        MOV     BUF0+LCTXVC,BUF0+RMTXVC
1860 001116 012667 022142                MOV     (SP)+,BUF0+LCTXVC
1861 001122 016746 022362                MOV     BUF0+RMRXVC,-(SP)
1862 001126 016767 022126 022354        MOV     BUF0+LCRXVC,BUF0+RMRXVC
1863 001134 012667 022120                MOV     (SP)+,BUF0+LCRXVC
1864 001140 042767 000020 32             BIC     #INITLD,TCONTL ;PREVENT AUTO-RESTART
1865 001146 000412                        BR      CONBAS      ;CONTINUE STARTUP IN COMMON FLOW(V01-01)
1866
1867                        ;CONSOLE ROOT
1868                        ;SET UP DEVICE VECTORS
1869                        ;THEN START UP CONSOLE PROGRAM
1870
1871 001150 105767 037747'      CONSRT: TSTB    APTL0D      ;RUNNING UNDER APT-MANF?
1872 001154 001313                BNE     APTSRT      ;BR IF YES AND SWAP VECTORS(V01-00,EDIT A)
1873 001156 000406                BR      CONBAS
1874
1875 001160 105267 035600      CONBOT: INCB    SETSWH      ;FORCE SWITCH TRANSITION SET-UP BY CHKSWH
1876 001164 011600                MOV     (SP),R0      ;GET POWER UP/RESTART FLAG
1877 001166 012706 001000                MOV     #1000,SP      ;RESET STACK TO SENSIBLE VALUE.
1878 001172 010046                MOV     R0,-(SP)      ;PUT POWER UP/RESTART FLAG ON STACK
1879 001174 012700 023200'      CONBAS: MOV     #BUF0,R0      ;SET UP DEVICE VECTORS
1880 001200 005001                CLR     R1          ;START AT ADDRESS 0
1881 001202 012021                20$: MOV    (R0)+,(R1)+      ;LOAD INTERRUPT VECTORS AND PSW'S
1882 001204 105701                TSTB    R1          ;STOP AT 400 (LOW BYTE = 000)
1883 001206 001375                BNE     20$
1884
1885
1886                        ;ENABLE TERMINAL KEYBOARD AND PRINTER INTERRUPTS
1887                        ;THESE TWO ENABLES ARE NEVER CLEARED
1888 001210 005777 037760'      TST     @RBUF      ;CLEAR OUT KBD BUFFER
1889 001214 004767 001376      JSR     PC,ENLTIE      ;ENABLE LOCAL TERMINAL INT ENABLES
1890 001220                MOVTOPSW    #0
1891 001224 110467 021255      MOVB    R4,CNVTDN      ;PASS ALONG 'TO LOAD OR NOT TO LOAD WCS' PARAMETER

```

ZZ-ESKAA-10.1 LOAD CONSOLE PROGRAM
V10-01-L MACRO V05.03 I ay 25-Apr-86 10:56 Page 11-2
LOAD CONSOLE PROGRAM

F 3

20-MAY-1986

Fiche 1 Frame F3

Sequence 31

1892 001230 004767 030572

JSR PC,CHKSWH

;SET-UP AS DIRECTED BY CONSOLE MODE SWITCH

```
1894 .SBTTL
1895 .SBTTL COMMAND GETTER
1896
1897 .ENABL LSB
1898
1899
1900 001234 RESTRT: T$INIT ;INIT TERMINALHANDLER
1901 001236 004767 001474 JSR PC,SETINP ;SET UP A 'WATCH-DOG' INPUT
1902 001242 012767 006416* 034146 MOV #DOSHOW,WHATTDO ;ENABLE A 'SHOW' COMMAND
1903 001250 004767 003170 JSR PC,SETTXR ;SET 'TX READY'
1904 001254 005037 173014 CLR #RXDONE ;CLEAR RX DONE
1905 001260 005067 021236 CLR RELJCA
1906 001264 005067 021234 CLR RELOCA+2 ;CLEAR RELOCATION REGISTER
1907 001270 005067 034326 CLR FLAG ;CLEAR CONTROL FLAGS
1908 001274 105767 037747* TSTB APTLOD ;DID APT LOAD US?
1909 001300 001055 BNE 10$ ;BR IF YES, SKIPPING LOAD AND VERSION CHECKS
1910 001302 004767 001524 JSR PC,SHOWIN ;PERFORM A 'SHOW' COMMAND, AND TEST FOR HALT
1911 001306 105767 021173 TSTB CNVTDN ;TEST FOR WCS-ECO LOAD
1912 001312 001441 BEQ 5$ ;BR IF NO LOAD TO DO
1913 001314 021627 123456 CMP (SP),#123456 ;HERE AS A RESULT OF POWER-UP?
1914 001320 001436 BEQ 5$ ;BR IF NOT(CRASH RECOVERY)
1915 001322 105067 037745* CLRB WRMSTR ;CLEAR WARM-START FLAG
1916 001326 105067 037746* CLRB CLDSTR ;CLEAR COLD-START FLAG
1917 001332 TYPEMES #WCSLOD,CR ;TELL OPERATOR WE ARE LOADING WCS
1918 001340 012700 017202* MOV #ECONAM,R0 ;SET UP TO OPEN ECO FILE
1919 001344 004767 015570 JSR PC,SETFIL ;MOVE ECO NAME TO FILENAME BLOCK
1920 001350 004767 010050 JSR PC,GETVER ;GET FPLA VERSION AND MICRO CODE OPTIONS
1921 ;(THIS CALL ADDED SO 'DOLOAD' KNOWS HOW MANY
1922 ;BYTES TO LOAD. VER 5-02)
1923 001354 052767 100000 034016 BIS #WCSDES,TCONTL ;MARK THE LOAD FOR WCS
1924 001362 012767 012322* 034026 MOV #DOLOAD,WHATTDO ;SET UP RTN POINTER
1925 001370 012767 010000 035206 MOV #FIRSTW,EFFADR ;SET BASE ADDRESS FOR LOAD
1926 001376 052767 000020 034216 BIS #NOSHOW,FLAG ;INHIBIT SHOWING VERSION ON THIS LOAD
1927 001404 004767 001422 JSR PC,SHOWIN ;EXECUTE THE LOAD
1928 001410 004767 010010 JSR PC,GETVER ;ASSEMBLE VERSION OF WCS,PCS, FPLA
1929 001414 000403 BR 9$ ;SHD VERSION
1930 001416 052767 004000 034176 5$: BIS #WCSPRES,FLAG ;DID NOT LOAD WCS BUT MARK IT PRESENT
1931 ;BECAUSE THIS IS A CRASH RECOVERY
1932 001424 004767 005336 9$: JSR PC,DOSHVR ;DISPALY VERSION INFO
1933 001430 004767 007670 JSR PC,TSTVER ;CHECK FOR VERSION COMPATIBILITY
1934 001434 004767 000140 10$: JSR PC,GETLIN ;GET A COMMAND LINE
1935 001440 012700 035400* MOV #TCONTL,R0
1936 001444 012701 000007 MOV #7,R1 ;SET UP TO CLEAR 7 WORDS IN A ROW
1937 001450 005020 15$: CLR (R0)+ ;CLEAR TCONTL,MICFLG,NEXTCT,COUNT,COUNT+2,DEEXBY(BYTE)
1938 001452 005301 DEC R1 ;DEFSTP(BYTE),ABORT(BYTE),AND RPTFLG(BYTE)
1939 001454 003375 BGT 15$
1940 001456 012720 003102* MOV #RTSINS,(R0)+ ;PRESET NULL COMMAND IN 'WHATTDO'
1941 001462 012701 177777 MOV #177777,R1 ;R0 NOW POINTING TO 'CURRAD'
1942 001466 010120 MOV R1,(R0)+ ;MARK CURRENT RADIX AND DATA LENGTH UNUSED
1943 001470 110120 MOV R1,(R0)+ ;MARK CURRENT ADDRESS SPACE UNUSED
1944 001472 012704 036421* MOV #TTYBUF+1,R4 ;POINT R4 TO INPUTTED COMMAND LINE
1945 001476 012705 015426* MOV #MAJTREE,R5 ;POINT R5 TO MAIN SENTENCE TREE
1946 001502 010503 MOV R5,R3 ;DITTO R3
1947 001504 004767 012616 JSR PC,RECOG ;TRY TO RECOGNIZE INPUT STRING, IF RECOGNIZED EXECUTE
1948 001510 103351 BCC 10$ ;BR IF COMMAND RECOGNIZED AND EXECUTED
```


1949	001512			TYPEMES	#CRMESQ,,CR	;TYPE FIRST PART OF ERROR MESSAGE
1950	001520	012701	036420'	MOV	#TTYBUF,R1	;POINT R1 TO BEGINNING OF CCMAND STRING
1951	001524	112100		MOVB	(R1)+,R0	;R0 GETS LENGTH OF INPUTTED COMMAND LINE
1952	001526	012746	021672'	MOV	#ISINCO,-(SP)	;ASSUME ONE ERROR MESSAGE WILL BE TYPED
1953	001532	121427	000015	CMPB	(R4),#15	;TEST FOR 'EOL' CAUSING ERROR
1954	001536	001410		BEQ	20\$;BR IF 'EOL' CAUSING ERROR
1955	001540	121427	000041	CMPB	(R4),#!	; ! IS ALSO AN 'EOL'
1956	001544	001405		BEQ	20\$;BR IF 'EOL' CAUSING ERROR
1957	001546	060100		ADD	R1,R0	;POINT R0 TO END OF COMMAND LINE
1958	001550	160400		SUB	R4,R0	;R0 GETS # OF CHARACTERS IN BAD PART OF STRING
1959	001552	010401		MOV	R4,R1	;R1 GETS POINTER TO BEGINNING OF BAD PART
1960	001554	012716	021653'	MOV	#ISANER,(SP)	;CHANGE THE MESSAGE WE GUESSED AT
1961	001560			20\$: TYPE	R1,R0	;R1 IS ADDRESS OF STRING,R0 IS LENGTH
1962	001574			TYPEMES		;TYPE THE ERROR MSG WHOSE ADDRESS IS ON STACK
1963	001576	000716		BR	10\$;GET ANOTHER LINE
1964						
1965					.DSABL	LSB

```

1967 .SBTTL GET A COMMAND LINE
1968
1969 .ENABL LSB
1970
1971 001600 GETLIN:
1972 001600 012702 035622' 1$: MOV #FLAG,R2
1973 001604 105767 034360 TSTB BOOTFL ;BOOTING?
1974 001610 001402 BEQ 6$ ;BR IF NO
1975 001612 000167 000510 14$: JMP 5$
1976
1977 001616 6$: ;+
1978 ;*****
1979 ;
1980 ; CHECK FOR AUTO-RESTART CONDITIONS
1981 ;
1982 ;*****
1983 ;-
1984 001616 032767 000020 033554 BIT #INITLD,TCONTL ;AUTO-RESTART FLAG SET?
1985 001624 001537 BEQ 22$ ;BR IF NO
1986 001626 042767 000020 033544 BIC #INITLD,TCONTL ;CLEAR AUTO-RESTART BIT
1987 001634 105767 037751' TSTB AUTFLG ;AUTO-RESTART ENABLED?
1988 001640 001526 BEQ 21$ ;BR IF NO
1989
1990 ;+
1991 ; BEGIN V01-EDIT-25
1992 001642 032712 000100 BIT #WFDONE,(R2) ;IS CONSOLE COMMAND HANDLER IN 'WAIT FOR DONE' STATE?
1993 001646 001403 BEQ 95$ ;BR IF NOT AND CONTINUE AUTO RESTART CHECKS
1994 001650 032712 020000 BIT #SFWDON,(R2) ;WAS A 'DONE' RECEIVED?
1995 001654 001065 BNE 9$ ;BR IF YES, ABORTING AUTO-RESTART
1996
1997 ;
1998 ; END V01-EDIT-25
1999 001656 105767 037745' 95$: TSTB WRMSTR ;WARM-START FLAG SET? (EDIT-16, PARTIAL)
2000 001662 001353 BNE 14$ ;SECOND TIME AROUND -- GO DO A BOOT
2001
2002 ;+
2003 ;*****
2004 ;
2005 ; AUTO-RESTART CONDITIONS ARE SATISFIED.
2006 ;
2007 ;
2008 ; PUT VAX PROGRAM COUNTER IN VAX GEN. REG. 10
2009 ; PUT VAX PSL IN VAX GEN. REG. 11
2010 ; PUT ERROR CODE (HALT REASON) IN VAX GEN. REG. 12
2011 ; THEN EXECUTE THE CONSOLE 'a' FILE NAMED 'RESTAR.CMD'.
2012 ;*****
2013 ;-
2014 001664 105267 037745' INCB WRMSTR ;SET FLAG TO AVOID INFINITE LOOP (END EDIT-16)
2015 001670 001670 TYPMES #AUTRES,,CR ;TYPE '(AUTO-RESTART)'
2016 001676 005001 CLR R1
2017 001700 012746 001712' MOV #7$,-(SP) ;STACK A RETURN FOR 'SETUPR' CALL
2018 001704 004067 015540 JSR R0,SETUPR ;SET UP ADDRESS AND ADDRESS SPACE FOLLOWING
2019 001710 002 012 .BYTE GENSPC,10. ;(SETS UP ACCESS TO GEN REG 10.)
2020 001712 105267 033474 7$: INCB DEEXBY ;FORCE DEPOSIT
2021 001716 005067 033462 CLR NEXTCT ;FORCE ONLY ONE

```

```

2022 001722 012701 036542'      MOV      #DATAFR,R1      ;SET TO MOVE 'DATAFR'(HALT PC) TO 'DATATO'
2023 001726 004767 000702      JSR      PC,MOVTOD      ;MOVE 'DATAFR' TO 'DATATO'
2024 001732 004767 002556      JSR      PC,DODEEX      ;DEPOSIT 'DATATO' TO GEN REG 10.(HALT PC)
2025 001736 012703 036562'      MOV      #DATATO,R3      ;SET TO READ PSL TO 'DATATO'
2026 001742 012702 000017      MOV      #17,R2      ;PSL IS ID 17(F)
2027 001746 004767 007070      JSR      PC,READID      ;READ PSL TO DATATO
2028 001752 004767 002536      JSR      PC,DODEEX      ;STORE 'DATATO' TO REG 11 (PSL)
2029 001756 012701 022532'      MOV      #SAVCO,R1      ;POINT R1 TO THE 'HALT REASON' CODE
2030 001762 004767 000646      JSR      PC,MOVTOD      ;MOVE 'SAVCO'(HALT REASON) TO 'DATATO'
2031 001766 004767 002522      JSR      PC,DODEEX      ;DEPOSIT 'DATATO' TO REG 12 (HALT REASON)
2032 001772 012700 017210'      MOV      #RESNAM,R0      ;R0 POINTS TO INDIRECT FILE NAME IN RAD50
2033 001776 004767 015136      JSR      PC,SETFIL      ;MOVE FILENAME BLOCK TO 'FILENM'
2034 002002 012767 003132' 033406 MOV      #DOAUTR,WHATTOD ;SET UP TO EXECUTE AN INDIRECT FILE
2035 002010 004767 001016      JSR      PC,SHOWIN      ;EXECUTE AN INDIRECT COMMAND FILE
2036 002014 000671
2037
2038 002016      8$:      ;+
2039      ;*****
2040      ;
2041      ;INDIRECT COMAND MODE. SET UP A TTY INPUT TO ALLOW THE OPERATOR
2042      ;TO ABORT THIS PROCESS VIA CONTROL-C IF DESIRED.
2043      ;
2044      ;*****
2045      ;-
2046 002016 004767 000714      JSR      PC,SETINP      ;SET UP AN INPUT
2047      ;+
2048      ;*****
2049      ;
2050      ;NOW WE CHECK FOR A WAIT IN PROGRESS(VIA A 'WAIT DONE' COMMAND).
2051      ;IF A WAIT IS IN PROGRESS , THEN WE DO NOT GET NEXT COMMAND LINE
2052      ;FROM THE INDIRECT COMMAND FILE UNTIL A 'DONE' CONDITION IS SENSED.
2053      ;
2054      ;*****
2055      ;-
2056 002022 032712 000100      BIT      #WFDONE,(R2)
2057 002026 001427      BEQ      15$      ;BR IF NO WAIT IN PROGRESS
2058 002030      9$:      ;+
2059      ;*****
2060      ;
2061      ;WAIT IN PROGRESS. HANG HERE IN A LOOP UNTIL: 1) CPU HALTS, OR 2) 'SFWDON' FLAG SETS
2062      ;NOTE: 'SFWDON' GETS SET BY 1) SIGNAL FROM VAX MACRO-LEVEL SOFTWARE, OR
2063      ;2)A CONTROL-C TYPED ON CONSOLE KEYBOARD. 'SFWDON' IS CLEARED EACH
2064      ;TIME THE STAR CPU IS STARTED OR CONTINUED.
2065      ;
2066      ;*****
2067      ;-
2068 002030 004767 006106      JSR      PC,TSTHAL      ;TEST FOR VAX CPU HALT
2069 002034 032712 020002      BIT      #SAWHLT!SFWDON,(R2) ;DID VAX SEND A 'DONE' OR HALT?
2070 002040 001773      BEQ      9$      ;BR IF NEITHER
2071 002042 032712 020000      BIT      #SFWDON,(R2)      ;WAS IT A 'SOFTWARE DONE' FROM VAX?
2072 002046 001406      BEQ      81$      ;BR IF NOT(HALTED WITHOUT SENDING 'DONE')
2073 002050 042767 000020 033322 BIC      #INITLD,TCONTL      ;INHIBIT AUTO-RESTART IN CASE VAX HALTED
2074 002056 042712 000100      BIC      #WFDONE,(R2)      ;DISABLE 'WAIT FOR DONE' MODE
2075 002062 000646
2076

```

```

2077 002064      81$:      ;+
2078              ;*****
2079              ;
2080              ;CPU HALTED WHILE WAITING FOR 'DONE' FROM MACRO-PROGRAM
2081              ;TYPE "<@EXIT>" AND ENABLE COMMAND LINE INPUT FROM TERMINAL.
2082              ;
2083              ;*****
2084              ;-
2085 002064 042712 000200      89$:      BIC      #INDMOD,(R2)      ;DISABLE INDIRECT MODE
2086 002070 012701 022373'      MOV      #INDEX1,R1      ;R1 GETS POINTER TO '<@EXIT>'
2087 002074 004767 011404      JSR      PC,INDECH      ;PRINT MESSAGE IF NOT BOOTING
2088 002100 105067 020405      CLRB     NOECHO      ;KILL ECHO SUPPRESSION
2089 002104 000635              BR       1$      ;TYPE PROMPT AND THEN ACCEPT INPUT FROM TERMINAL
2090
2091 002106      15$:      ;+
2092              ;*****
2093              ;
2094              ;GET AN INDIRECT COMMAND LINE FROM A FLOPPY FILE
2095              ;
2096              ;*****
2097              ;-
2098 002106 004767 011146      JSR      PC,INDLIN      ;GET A COMMAND LINE FROM THE FLOPPY
2099 002112 103174              BCC      4$      ;BR IF LINE GOTTEN WITH NO ERROR
2100 002114 000765              BR       89$      ;ERROR OR EOF ON INDIRECT FILE.
2101
2102 002116      21$:      ;+
2103              ;*****
2104              ;
2105              ;CHECK FOR: 'PROGRAM I/O MODE', 'INDIRECT COMMAND MODE'
2106              ;      'TALK MODE' OR 'CONSOLE MODE'
2107              ;
2108              ;IF<PROGRAM I/O MODE 'OR' TALK MODE> THEN <ENTER CONSOLE NULL LOOP>
2109              ;IF<INDIRECT MODE> THEN <GET A COMMAND LINE FROM A FLOPPY FILE>
2110              ;IF<NONE OF THE ABOVE> THEN<ISSUE REQUEST FOR TERMINAL INPUT, ENTER
2111              ;      ENTER CONSOLE NULL LOOP>
2112              ;
2113              ;*****
2114              ;-
2115 002116 042767 000020 033254      22$:      BIC      #INITLD,TCONTL      ;CLEAR SOFT AUTO-RESTART BIT
2116 002124 105067 033466      CLRB     LINGOT      ;CLEAR LINE SYNC FLAG
2117 002130 105712              TSTB     (R2)      ;TEST FOR INDIRECT COMMAND MODE
2118 002132 100731              BMI      8$      ;BR IF INDIRECT MODE
2119 002134 105767 033455      TSTB     PGM10M      ;TEST FOR PROGRAM I/O MODE
2120 002140 001033              BNE      NULJOB      ;BR IF PROGRAM I/O MODE
2121 002142              T$INIT      ;CANCEL ANY EXISTING READ REQUEST
2122 002144 032767 000000 033360      BIT      #TLKMOD,TCTFLG      ;IN TALK MODE?
2123 002152 001026              BNE      NULJOB      ;BR IF YES
2124 002154 105067 034241      CLRB     TTYBUF+1      ;MAKE SURE FIRST CHARACTER IS NOT 'X'
2125 002160              T$READ     #TTYBUF,#80..,GOTLIN      ;ISSUE REQUEST FOR TERMINAL INPUT
2126 002200 103003              BCC      80$      ;BR IF NO ERROR ON READ REQUEST
2127 002202 005726              TST      (SP)+      ;GET ERROR CODE OFF STACK
2128 002204 000167 177370      30$:      JMP      1$
2129
2130 002210 012746 022414'      80$:      MOV      #CONPHP,-(SP)      ;ASSUME NORMAL PROMPT
2131 002214 105767 020272      TSTB     LINKNG      ;ARE WE LINKING?

```

ZZ-ESKAA-10.1 GET A COMMAND LINE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 13-3
GET A COMMAND LINE

L 3

20-MAY-1986

Fiche 1 Frame L3

Sequence 37

2132 002220 001402
2133 002222 012716 022422'
2134 002226

23\$: BEQ 23\$
MOV #LNKPMP,(SP)
TYPEMES

;BR IF NO
;CHANGE TO REVERSE PROMPT FOR LINK
;TYPE A PROMPT(ADDRESS ON STACK)

```

2136 .SBTTL CONSOLE NULL LOOP
2137
2138 002230 NULJOB: .CONSOLE NULL LOOP
2139 002230 004767 027572 JSR PC,CHKSWH ;CHECK FOR CONSOLE MODE SWITCH CHANGE
2140 002234 004767 005702 JSR PC,TSTHAL ;WATCH FOR CPU HALTS
2141 002240 103761 BCS 30$ ;BR IF HALT SEEN
2142 ;*****EDIT-27 21-JUL-78*****
2143 ;DON'T CHECK FOR TIMEOUT UNTIL THE HARDWARE IS ECO'D CORRECTLY
2144 ;
2145 ;JSR PC,TSTTMO ;TEST FOR A 'MICRO-MACHINE TIME OUT'
2146 ;BCS 30$ ;BR IF TIMEOUT
2147 ;*****
2148 002242 004767 005346 JSR PC,TSTCLK ;TEST FOR CLOCK STOP
2149 002246 103756 BCS 30$ ;BR IF CLOCK STOPPED
2150 002250 012701 173034 MOV #MCS,R1
2151 002254 005367 033716 DEC FLPTIM ;BUMP FLOPPY POWER-OFF TIMER
2152 002260 001067 BNE 35$ ;BR IF FIRST WORD NOT UNDEFFLOWED
2153 002262 004767 000330 JSR PC,ENLTIE ;ENABLE LOCAL TERMINAL INTERRUPTS
2154 002266 005367 033706 DEC FLPTIM+2 ;BUMP HIGH ORDER BITS
2155 002272 001092 BNE 35$ ;BR IF NOT TIMED-OUT
2156 002274 042711 010000 BIC #FLPYOF,(R1) ;TURN OFF FLOPPY POWER
2157 002300 032711 004000 35$: BIT #BOOTBT,(R1) ;BOOT SWITCH ASSERTED?
2158 002304 001431 BEQ 3$ ;BR IF NO
2159 002306 052711 004000 40$: BIS #BOOTBT,(R1) ;CLEAR BOOT BIT(YES, A BIT SET!)
2160 002312 032711 004000 BIT #BOOTBT,(R1) ;SWITCH DE-ASSERTED?
2161 002316 001373 BNE 40$ ;BR IF NOT
2162 002320 032711 000003 BIT #<REMOT!LOCKD>,(R1) ;ANY MODE EXCEPT LOCAL?
2163 002324 001021 BNE 3$ ;BR IF NOT IN 'LOCAL' MODE
2164 002326 105767 037746 5$: TSTB CLDSTR ;COLD-START FLAG SET (EDIT-16 PARTIAL)
2165 002332 001013 BNE 97$ ;DON'T BOOT -- KEEP LOOPING
2166 002334 T$INIT ;CANCEL TERMINAL REQUEST
2167 002336 TYPEMES #BOTING,,CR ;TELL THE WORLD WE ARE BOOTING
2168 002344 012767 003104 033044 MOV #DOBOOT,WHATTODO ;SET UP BOOT COMMAND VECTOR
2169 002352 004767 014556 JSR PC,STBOFL ;SET UP BOOT FILE NAME
2170 002356 004767 000450 JSR PC,SHOWIN ;PERFORM THE BOOT
2171 002362 105067 033602 97$: CLRB BOOTFL ;CLEAR THE BOOT FLAG
2172 002366 000706 BR 30$
2173
2174 002370 105767 033222 3$: TSTB LINGOT ;TEST LINE SYNC FLAG
2175 002374 001715 BEQ NULJOB ;BR IF LINE NOT INPUTTED
2176 002376 100425 BMI 55$ ;BR IF ERROR ON LINE INPUT
2177 002400 105767 020106 TSTB LINKNG ;LINKING COMMANDS?
2178 002404 001437 BEQ 4$ ;BR IF NOT
2179 002406 016700 020144 MOV INDBYT,R0 ;POINT R0 TO BUFFER
2180 002412 012701 036420 MOV #TTYBUF,R1 ;POINT F1 TO INPUT LINE
2181 002416 112102 MOVB (R1)+,R2 ;R2 GETS LENGTH OF LINE
2182 002420 112103 50$: MOVB (R1)+,R3 ;R3 GETS BYTE TO XFER
2183 002422 004767 000064 JSR PC,SAVBTE ;SAVE BYTE IN R3
2184 002426 103414 BCS 60$ ;BR IF WRITE ERROR OR OVERFLOW
2185 002430 005302 DEC R2 ;ALL BYTES XFERRRED?
2186 002432 003372 BGT 50$ ;BR IF NO
2187 002434 012703 000012 MOV #12,R3 ;PUT A LINE FEED CHAR IN R3
2188 002440 004767 000046 JSR PC,SAVBTE ;PUT LINEFEED AT END OF LINE
2189 002444 010067 020106 MOV R0,INDBYT ;RESET BUFFER POINTER
2190 002450 103255 BCC 30$ ;BR IF NO WRITE ERROR OR OVERFLOW

```

```
2191 002452 105767 020034      55$: TSTB LINKNG ;LINKING COMMANDS?
2192 002456 001652              BEQ 30$ ;BR IF NOT
2193 002460 105067 020026      60$: CLRB LINKNG ;TERMINATE LINKING
2194 002464 016700 020066      MOV INDBYT,R0 ;POINT R0 TO BUFFER (V01-EDIT-26)
2195 002470 005003              CLR R3 ;WRITE A BLANK AT END OF BUFFER
2196 002472 004767 000014      JSR PC,SAVBTE ;PUT R3 IN BUFFER
2197 002476 004767 000020      JSR PC,FORCWT ;FORCE OUT BUFFER
2198 002502 000640              BR 30$
2199
2200 002504 105367 033710      4$: DECB TTYBUF ;COMPENSATE FOR CR AT END OF LINE
2201 ;SET UP A 1 CHARACTER INPUT BUFFER TO WATCH FOR CONTROL-C WHILE
2202 ;CONSOLE IS EXECUTING THE COMMAND
2203 002510 000512              BR SETINP
2204
2205 .DSABL LSB
```

ZZ-ESKAA-10.1 CONSOLE NULL LOOP
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 15
 CONSOLE NULL LOOP

```

2207      .ENABL  LSB
2208 002512 SAVBTE: ;SAVE A BYTE OF COMMAND LINE FOR LINKING
2209      ;INPUTS:      R3 IS BYTE TO XFER
2210      ;              R0 IS POINTER TO CURRENT BYTE OF BUFFER
2211      ;              'INDLFT' IS # OF SECTORS REMAINING FOR LINK
2212      ;              'INDSEC' IS CURENT SECTOR FOR LINK
2213      ;
2214      ;OUTPUTS:      R0<--R0+1(BUFFER NOT FULL) OR R0<--'BUF0 (BUF FULL)
2215      ;              C BIT SET IF ERROR OR OVERFLGW
2216
2217 002512 110320      MOVB      R3,(R0)+      ;PUT BYTE IN BUFFER
2218 002514 020027 023400*      CMP      R0,#BUF0+128.      ;OVER BUFFER?
2219 002520 002433      BLT      10$      ;BR IF NOT
2220 002522 012700 023200*      FORCWT: MOV      #BUF0,R0      ;RESET R0
2221 002526      F$WRIT  INDSEC,R0      ;WRITE CURRENT BUFFER
2222 002564 103004      BCC      5$      ;BR IF NO ERROR
2223 002566 012600      MOV      (SP)+,R0      ;R0 GETS ERROR CODE
2224 002570 004767 011050      JSR      PC,TYFLER      ;TYPE FLOPPY ERROR MSG
2225 002574 000406      BR      20$
2226
2227 002576 005267 017760      5$:      INC      INDSEC      ;UPDATE SECTOR #
2228 002602 005367 017752      DEC      INDLFT      ;MINUS ONE FROM # OF SECTORS PERMISSABLE
2229 002606 003401      BLE      20$      ;BR IF # SECTORS EXCEEDED
2230 002610 005727      10$:      TST      (PC)+      ;CLEAR C BIT, SKIP NEXT INST
2231 002612 000261      20$:      SEC
2232 002614 000207      30$:      RTS      PC
2233      .DSABL  LSB
2234
2235 002616      ENLTIE: ;ENABLE LOCAL TERMINAL INTERRUPT ENABLES
2236      ;THIS RTN IS ENTERED PERIODICALLY TO INSURE THE CONSOLE
2237      ;DOESN'T GO DEAD IF THE LOCAL TERMINAL INTERRUPT ENABLES ARE
2238      ;CLEARED UNEXPECTEDLY
2239 002616 052777 000100 037762*      BIS      #XMTINT,@XCSR      ;ENABLE INTERRUPTS
2240 002624 052777 000100 037756*      BIS      #RCVINT,@RCR      ;ENABLE INTERRUPTS
2241 002632 000207      20$:      RTS      PC
2242
2243 002634      MOVTOD: ;MOVE DATA POINTED BY R1 TO 'DATATO'
2244 002634 012700 036562*      MOV      #DATATO,R0
2245 002640 012120      MOV      (R1)+,(R0)+
2246 002642 012120      MOV      (R1)+,(R0)+
2247 002644 000207      RTS      PC

```



```

2249      ;DONE VECTOR ENTRY FOR ONE CHARACTER INPUT ROUTINE
2250 002646 103014      GOTINP: BCC 1$      ;BR IF NO ERROR
2251 002650 026627 000002 000006      CMP 2(SP),#$TCTC      ;CHECK FOR CONTROL-C
2252 002656 001027      BNE SETINP      ;BR IF NOT CONTROL-C
2253 002660 105267 032530      INCB ABORT      ;SET ABORT
2254 002664 105067 032525      CLRB RPTFLG      ;DISABLE REPEAT MODE
2255 002670 052767 020000 032724      BIS #SFWDON,FLAG      ;TERMINATE A 'WAIT IN PROGRESS'
2256 002676 000414      BR 2$
2257
2258 002700 032767 001000 032714 1$: BIT #SPCSTP,FLAG      ;TEST FOR SPACE BAR STEP ENABLED
2259 002706 001413      BEQ SETINP      ;BR IF NOT ENABLED
2260 002710 126727 033503 000040      CMPB TTYTMP+1,#40      ;TEST FOR A SPACE INPUTTED
2261 002716 001004      BNE 2$      ;BR IF NOT A SPACE
2262 002720 052767 000400 032674      BIS #SPCSYC,FLAG      ;SET SYNC FLAG FOR DOSTEP ROUTINE
2263 002726 000403      BR SETINP
2264
2265 002730 042767 001600 032664 2$: BIC #SPCSTP!SPCSYC!INDMOD,FLAG ;DISABLE SPACE-BAR STEP
2266 002736      SETINP:;ROUTINE TO SET UP A ONE CHARACTER INPUT
2267 002736      T$READ #TTYTMP,#1,#GOTINP
2268 002756 103001      BCC 1$      ;BR IF NO ERROR
2269 002760 005726      TST (SP)+      ;CLEAR STACK OF ERROR CODE
2270 002762 000207      1$: RTS PC
2271

```

```
2273 .SBTTL
2274 .SBTTL COMMAND EXECUTER
2275
2276 .ENABL LSB
2277
2278 002764 EXECUT: ;EXECUTE THE COMMAND JUST PARSED
2279 ;INPUTS: 'WHATTDO' POINTS TO ROUTINE TO EXECUTE
2280 ; ALL COMMAND RELATED DATA SET UP BY PARSER
2281 ;
2282 ; ALL REGISTERS ARE VOLATILE
2283 ;
2284 ;OUTPUTS: NONE
2285 ;
2286 ;EFFECTS: IF<RPTFLG=0> THEN<COMMAND EXECUTED ONCE>
2287 ; IF<RPTFLG=1> THEN<COMMAND EXECUTED CONTINUOUSLY>
2288
2289
2290
2291 ;SEQUENCE OF ACTION:
2292 ; 1) APPLY SWITCHES OR DEFAULTS FOR RADIX,ADDRESS SPACE, DATA LENGTH
2293 ; 2) EXECUTE COMMAND
2294 ; 3) TEST FOR REPEAT
2295
2296 002764 012700 000003 MOV #3,R0
2297 002770 012701 035420' MOV #CURRAD,R1 ;POINT R1 TO CURRENT RADIX BYTE
2298 002774 105067 017510 CLRB TMPRAD ;USE TEMRAD AS A FLAG FOR 'DOSTDF'
2299 003000 105721 10$: TSTB (R1)+ ;TEST FOR SWITCH ON RADIX,ADDRESS SPACE, OR LENGTH
2300 003002 100004 BPL 20$ ;BR IF A SWITCH WAS APPLIED
2301 003004 116161 000002 177777 MOVB 2(R1),-1(R1) ;MOVE DEFAULT TO CURRENT USAGE BYTE
2302 003012 000402 BR 25$
2303
2304 003014 105367 017470 20$: DECB TMPRAD ;NOTE THAT AT LEAST ONE QUALIFIER APPLIED
2305 003020 005300 25$: DEC R0
2306 003022 003366 BGT 10$
2307 003024 105767 032364 30$: TSTB ABORT ;TEST FOR COMMAND ABORT
2308 003030 001022 BNE 40$ ;BR IF ABORT SET
2309 003032 012700 035400' SHOWIN: MOV #1CONTL,R0
2310 003036 005001 CLR R1
2311 003040 012702 011124' MOV #TSTRUN,R2 ;USEFUL POINTER FOR MANY COMMANDS
2312 003044 012703 035622' MOV #FLAG,R3 ;DITTO
2313 003050 012704 173032 MOV #MCR,R4
2314 003054 012705 173034 MOV #MCS,R5
2315 003060 004777 032332 JSR PC,@WHATTDO ;PERFORM COMMAND
2316 003064 004767 005052 JSR PC,TSTHAL ;TEST FOR A HALT
2317 003070 105767 032321 TSTB RPTFLG ;TEST FOR REPEAT
2318 003074 001353 BNE 30$ ;BR IF REPEAT IS SET
2319 003076 012703 016132' 40$: MOV #MTEOL,R3 ;WILL CAUSE 'RECOG' TO QUIT PARSING
2320 003102 000207 RTSINS: RTS PC
2321
2322 .DSABL LSB
```

```
2324 .SBTTL COMMAND EXECUTION RTN REGISTER USAGE SUMMARY
2325
2326 ;ALL OF THE FOLLOWING ROUTINES IN THIS MODULE CALLED 'DOXXXX'
2327 ;ARE ENTERED BY THE ROUTINE CALLED 'EXECUT'
2328 ;
2329 ;
2330 ;THE GENERAL REGISTERS ARE PRESET BY 'EXECUT' AS FOLLOWS:
2331 ; R0-->'TCONTL'
2332 ; R1 IS CLEAR (0)
2333 ; R2-->'TSTRUN'
2334 ; R3-->'FLAG'
2335 ; R4--> MCR REGISTER
2336 ; R5--> MCS REGISTER
```

```

2338 .SBTTL BOOT,PROCESS INDIRECT FILE,CLEAR SOMM,CONTINUE
2339
2340 .ENABL LSB
2341
2342 003104 DOBOOT: ;PERFORM A BOOT SEQUENCE
2343 ;R2-->'TSTRUN'
2344 003104 004767 006214 JSR PC,,STVER ;INSURE VERSION COMPATIBILITY BETWEEN PCS,WCS,FPLA
2345 003110 103442 BCS 20$ ;BR IF FATAL INCOMPATIBILITY
2346 003112 004712 JSR PC,(R2) ;TEST FOR STAR RUNNING
2347 003114 103440 BCS 20$ ;BR IF SO
2348 003116 105267 037745' INCB WRMSTR ;SET WARM-START AND COLD-START FLAGS
2349 003122 105267 037746' INCB CLDSTR ; WHENEVER TRYING TO BOOT (EDIT-16)
2350 003126 004767 001212 JSR PC,STRCK ;START CPU CLOCK
2351 003132 105267 017353 DOAUTR: INCB NOECHO ;SET 'NO ECHO' FLAG(AUTO-RESTART ENTRY)
2352 003136 000402 BR 5$
2353
2354 003140 DOINDI: ;OPEN AN INDIRECT COMMAND FILE
2355 ;R2-->'TSTRUN'
2356 ;R3-->'FLAG'
2357 003140 105067 017345 CLR B NOECHO ;CLEAR 'NO ECHO' FLAG
2358 003144 5$: OPEN$ #FILENM ;OPEN FILE ON DRIVE 0 OR 1
2359 003154 103420 BCS 20$ ;BR IF OPEN FAILED
2360 003156 005067 017402 CLRSIB: CLR SECLD ;GUARANTEES A CHANGE IN FLOPPIES WON'T SCREW UP
2361 003162 012700 022564' LNKENT: MOV #INDSEC+2,R0 ;R0 GETS A LIST POINTER
2362 003166 012640 MOV (SP)+,-(R0) ;SAVE STARTING SECTOR OF FILE
2363 003170 012640 MOV (SP)+,-(R0) ;SAVE # OF SECTORS IN FILE
2364 003172 012740 023400' MOV #BUF0+128,-(R0) ;CAUSE FIRST SECTOR TO BE READ
2365 003176 052713 000200 BIS #INDMOD,(R3) ;ENABLE INDIRECT MODE
2366 003202 042713 000100 BIC #WFDONE,(R3) ;INIT 'WAIT FOR DONE' FLAG
2367 003206 000207 RTS PC
2368
2369 003210 DOCLSO: ;CLEAR SOMM ENABLE ON CPU INTERFACE BOARD
2370 ;R4-->MCR
2371 003210 042714 000100 BIC #SOMMB,(R4)
2372 003214 000241 10$: CLC
2373 003216 000207 20$: RTS PC
2374
2375 003220 DOCONT: ;PERFORM A STAR CPU CONTINUE
2376 ;R2-->TSTRUN
2377 ;R4-->MCR
2378 ;R5-->MCS
2379 003220 004767 006100 JSR PC,TSTVER ;INSURE VERSION COMPATIBILITY BETWEEN PCS,WCS, AND FPLA
2380 003224 103774 BCS 20$ ;BR IF FATAL INCOMPATIBILITY
2381 003226 004712 JSR PC,(R2) ;TEST FOR CPU RUNNING
2382 003230 103772 BCS 20$ ;EXIT IF CPU RUNNING
2383 003232 004767 005360 CONTSQ: JSR PC,TSTTY2 ;CLEAR OUT CODE 2 MICRO-ERRORS
2384 003236 012700 000447 MOV #CONCON,R0 ;R0 GETS ADDRESS OF 'CRO-CONTINUE
2385 003242 004767 005464 JSR PC,PUSHU ;PUSH R0 ON MICRO-STACK
2386 ;CLR R1 ;(IN 'PUSHU' RTN)
2387 003246 103763 BCS 20$ ;BR IF CLOCK STOPPED
2388 003250 106746 MFPS -(SP) ;**
2389 003252 106427 000340 MTPS #340 ;BLOCK OUT LSI INTERRUPTS
2390 003256 005767 032340 TST FLAG ;TEST FOR SINGLE INST MODE
2391 003262 100402 BMI 30$ ;BR IF SINGLE INST
2392 003264 042714 100000 BIC #HLTREQ,(R4) ;CLEAR HALT REQUEST BIT ON CIB

```

```

2393 003270 012702 173014      30$:  MOV    #RXDONE,R2      ;RESTORE THE ID BUS INTERFACE TO STARLET
2394 003274 005022              CLR    (R2)+          ;CLEAR 'RX DONE'
2395 003276 005012              CLR    (R2)           ;INSURE 'TX READY' STARTS OUT CLEAR
2396 003300 052722 000200      BIS    #TXRDY,(R2)+    ;SET 'TX READY'
2397 003304 032767 000004 032310  BIT    #IDSAVD,FLAG    ;TEST FOR 'TOLD' REGISTER SAVED
2398 003312 001407              BEQ    40$             ;BR IF NOT SAVED
2399 003314 016722 033256      MOV    SAVIDL,(R2)+    ;RESTORE 'TOLD' REGISTER
2400 003320 016722 033254      MOV    SAVIDH,(R2)+
2401 003324 052737 000200 173014  BIS    #RXDNE,@#RXDONE ;SET 'RX DONE'
2402 003332 042767 020002 032262 40$:  BIC    #SAWHLT!SFWDON,FLAG ;CLEAR 'HALT SEEN' AND 'SOFTWARE DONE'
2403 003340 105767 032051      TSTB   RPTFLG         ;IS REPEAT SET? (PX0102)
2404 003344 001005              BNE    50$             ;BR IF IT IS AND SKIP SETTING P I/O
2405 003346 105267 032243      INCB   PGM10M         ;ENABLE PROGRAM I/O MODE
2406 003352 042767 100400 032152  BIC    #ROFLAG!PRNINH,TCTFLG ;CLEAR PRINT-INHIBIT, RUBOUT SERVICE
2407 003360 042714 000400      50$:  BIC    #STRIND,(R4)      ;ENABLE STAR INTERRUPTS FROM RX AND TX
2408 003364 052715 000140      BIS    #RDYIE!DNEIE,(R5) ;ENABLE LSI INTS FROM 'TXRDY' AND 'RXDNE'
2409 003370 052714 002000      BIS    #MAINTR,(R4)    ;POP MICRO-STACK
2410 003374 106426              MTPS   (SP)+          ;*8*
2411                          ;AT THIS POINT WE WILL INTERRUPT OUT OF HERE IF THERE
2412                          ;IS A CHARACTER TO GO TO THE STAR CPU VIA 'TOLD'
2413 003376 000706              BR     10$             ;EXIT
2414
2415 003400      DOPERF: ;PROCESS A FILE OF LINKED COMMANDS
2416 003400 012746 000012      MOV    #10,-(SP)        ;STACK MAX # OF SECTORS IN LINK FILE
2417 003404 012746 000016      MOV    #14,-(SP)        ;STACK STARTING LOGICAL SECTOR(LOG BLK 3)
2418                          ;*
2419                          ; BEGIN EDIT-26 V01
2420                          ;
2421                          ; THE FOLLOWING FIX ALLOWS LINK FILES OF LESS THAN 128 BYTES
2422                          ; TO BE EXECUTED CONTINUOUSLY, WITHOUT HAVING TO RE-READ THE
2423                          ; LINK FILE ON EACH RE-EXECUTION. BASICALLY, THE LOGIC OF
2424                          ; THIS FIX IS AS FOLLOWS:
2425                          ; IF<EXECUTING A PERFORM, 'AND' CONSOLE ALREADY IN 'a' MODE>
2426                          ; THEN<INHIBIT CLEARING THE 'CURRENT-SECTOR-IN-BUFFER' VARIABLE>
2427
2428 003410 032713 000200      BIT     #INDMOD,(R3)    ;IS CONSOLE IN INDIRECT MODE?
2429 003414 001262              BNE    LNKENT         ;BR IF YES. DO NOT CLEAR 'CURRENT SECTOR' VARIABLE
2430 003416 000657              BR     CLRSIB         ;CLEAR THE 'CURRENT SECTOR IN BUFFER' VARIABLE SO
2431                          ; THAT THE FLOPPY WILL BE READ REGARDLESS OF THE
2432                          ; SECTOR # THAT IS ALREADY IN THE BUFFER.
2433                          ;
2434                          ; END OF EDIT-26 V01
2435                          ;-
2436                          .DSABL LSB

```

```

2438 .SBTTL START,UNJAM
2439
2440 .ENABL LSB
2441
2442 003420 DOSTAR: ;PERFORM A STAR CPU START(INIT,DEPOSIT PC, CONTINUE)
2443 ;R0-->'TCONTL'
2444 ;R2-->'TSTRUN'
2445 ;R4-->'MCR'
2446 003420 042710 000020 BIC #INITLD,(R0) ;CLEAR AUTO-RESTART FLAG
2447 003424 004767 005674 JSR PC,TSTVER ;INSURE COMPATIBILITY BETWEEN PCS,WCS, AND TPLA
2448 003430 103421 BCS 10$ ;BR IF FATAL INCOMPATIBILITY
2449 003432 004712 JSR PC,(R2) ;TEST FOR CPU RUNNING
2450 003434 103417 BCS 10$ ;BR IF CPU RUNNING
2451 003436 016767 033142 033116 MOV EFFADR,DATATO ;PUT EFFECTIVE ADDRESS INTO DEPOSIT DATA AREA
2452 003444 016767 033136 033112 MOV EFFADR+2,DATATO+2
2453 003452 005710 TST (R0) ;TEST FOR A WCS START
2454 003454 100010 BPL 20$ ;BR IF NOT A WCS START
2455 003456 016700 033100 MOV DATATO,R0 ;R0 GETS ADDRESS TO START AT
2456 003462 004767 005244 JSR PC,PUSHU ;PUSH R0 ON MICRO-STACK
2457 003466 103402 BCS 10$ ;BR IF CLOCK STOPPED
2458 003470 052714 002000 BIS #MAINTR,(R4) ;POP MICRO-STACK
2459 003474 000207 10$: RTS PC
2460
2461 003476 004767 000130 20$: JSR PC,INITQU ;DO A STAR INIT AND CLEAR STARLET INPUT QUEUE
2462 ; (EDIT-21A)
2463 003502 004767 004212 JSR PC,CHAIT ;WAIT FOR INIT TO FINISH
2464 003506 103772 BCS 10$ ;EXIT IF TIME OUT
2465 003510 105267 031676 INCB DEEXBY ;FORCE A DEPOSIT
2466 003514 004767 002260 JSR PC,EXDEPC ;DEPOSIT 'DATATO' TO STAR PC
2467 003520 103765 BCS 10$ ;BR IF DEPOSIT FAILED
2468 003522 000643 BR CONTSQ ;DO A CONTINUE
2469
2470
2471 003524 DOUNJA: ;PERFORM AN SBI UNJAM
2472 ;R2-->TSTRUN
2473 ;R4-->MCR
2474 003524 004712 JSR PC,(R2) ;TEST FOR CPU RUNNING
2475 003526 103414 BCS 30$ ;BR IF RUNNING
2476 003530 012700 000452 MOV #SBIUNJ,R0 ;R0 GETS ADDRESS OF UNJAM SBI MICRO-RTN
2477 003534 004767 005172 JSR PC,PUSHU ;PUSH R0 ONTO MICRO-STACK
2478 003540 103407 BCS 30$ ;BR IF PUSH FAILED
2479 003542 052714 002000 BIS #MAINTR,(R4) ;POP MICRO-STACK TO MICRO-PC
2480 003546 COMWAT: ;WAIT FOR STAR CPU TO RESPOND, THEN TEST FOR ERRORS
2481 ;OUTPUTS: C BIT SET IF TIMEOUT OR ERROR
2482 003546 004767 004146 JSR PC,CHAIT ;WAIT FOR COMPLETION
2483 003552 103402 BCS 30$ ;BR IF WAIT TIMED OUT
2484 003554 004767 004176 JSR PC,TSTERR ;TEST FOR SUCCESS ON FUNCTION
2485 003560 000207 30$: RTS PC
2486
2487 .DSABL LSB

```

```

2489          .SBTTL  HALT,INITIALIZE
2490
2491          .ENABL  LSE
2492
2493 003562      DOHALT: ;PERFORM A STAR CPU HALT
2494                ;R4-->MCR
2495                ;R5-->MCS
2496 003562 105715  TSTB    (R5)          ;TEST FOR CPU ALREADY HALTED
2497 003564 100004  BPL     10$          ;BR IF NOT HALTED
2498 003566                TYPEMES #ALRDHA,,CR ;TELL OPERATOR ALREADY HALTED
2499 003574 000407  BR      20$          ;EXIT
2500
2501 003576 052714 100000 10$: BIS     #HLTREQ,(R4) ;REQUEST STAR TO HALT
2502 003602 004767 177740 JSR     PC,COMWAT ;WAIT FOR STAR TO HALT
2503 003606 103403 BCS     30$          ;SKIP HALT REPORT IF TIMEOUT
2504 003610 004767 004374 JSR     PC,REPHLT ;REPORT THE HALT
2505 003614 000241 20$: CLC
2506 003616 000207 30$: RTS     PC
2507
2508
2509 003620      DOINIT: ;PERFORM A STAR CPU INITIALIZE
2510                ;INITIALIZE PRIMITIVE
2511                ;R2-->'TSTRUN'
2512 003620 004712 JSR     PC,(R2)          ;TEST FOR CPU RUNNING
2513 003622 103775 BCS     30$          ;BR IF CPU IS RUNNING
2514 003624 004767 000002 JSR     PC,INITQU ;DO COMMON INITIALIZE SEQUENCE AND
2515                ; CLEAR STARLET INPUT QUEUE (EDIT-21A)
2516 003630 000746 BR      COMWAT          ;GO WAIT FOR STAR TO FINISH
2517
2518 003632      INITQU: ;INITIALIZE STARLET INPUT QUEUE (EDIT-21A)
2519                ; USED ON 'LOAD CONSOLE' , 'INIT' , 'START'
2520 003632 012767 036246' 032360 MOV     #QUEBGN,FILLP ;SET FILL POINTER TO BEGINNING OF QUEUE
2521 003640 012767 036246' 032354 MOV     #QUEBGN,EMPTYP ;RESET BUFFER EMPTY POINTER
2522 003646 105067 032352 CLR     QUECNT ;SET QUEUE COUNTER TO 0
2523                ; (END EDIT-21A)
2524
2525 003652      INITRT: ;COMMON INITIALIZE SEQUENCE
2526                ;R4-->MCR
2527 003652 012703 035622' MOV     #FLAG,R3
2528 003656 042713 000004 BIC     #IDSAVD,(R3) ;FORGET ABOUT SAVED ID BUS STATE
2529 003662 005067 032726 CLR     TBF0SV ;CLEAR TBUF0 SAVED STATE
2530 003666 005067 032724 CLR     TBF0SV+2
2531 003672 052714 000002 BIS     #SBC,(R4) ;STOP CPU CLOCK
2532 003676 052714 010000 BIS     #CPURES,(R4) ;ISSUE A CPU HARDWARE RESET
2533 003702 042714 000200 BIC     #ROMNOP,(R4) ;MAKE SURE ROM NOP IS CLEAR
2534 003706 004767 000432 JSR     PC,STRTCK ;RESTART CPU CLOCK
2535 003712 042714 010000 BIC     #CPURES,(R4) ;DEASSERT CPU RESET SIGNAL
2536 003716 042713 000040 BIC     #SAWERR,(R3) ;FORGET ABOUT ANY CODE 2 MICRO-ERRORS
2537 003722 052713 000002 BIS     #SAWHLT,(R3) ;INHIBIT REPORTING A HALT
2538 003726 000207 RTS     PC
2539
2540
2541          .DSABL  LSB

```

ZZ-ESKAA-10.1 NEXT(PERFORM A STEP)
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 22
 NEXT(PERFORM A STEP)

```

2543 .SBTTL NEXT(PERFORM A STEP)
2544
2545 .ENABL LSB
2546
2547 003730 DONEXT: ;PERFORM A STEP
2548 ;R3-->'FLAG'
2549 ;R4-->'MCR'
2550 003730 005767 031452 TST COUNT ;TEST FOR ENABLE SPACE-BAR-STEP MODE
2551 003734 003002 BGT 10$ ;BR IF STEP COUNT > 0
2552 003736 052713 001000 BIS #SPCSTP,(R3) ;ENABLE SPACE-BAR-STEP MODE
2553 003742 032714 000006 10$: BIT #STS!SBC,(R4) ;TEST FOR SINGLE BUS CYCLE OR TIME STATE MODE
2554 003746 001007 BNE 20$ ;BR IF EITHER
2555 003750 005713 TST (R3) ;TEST FOR SINGLE INST MODE
2556 003752 100405 BMI 20$ ;BR IF SINGLE INST
2557 003754 004767 000350 JSR PC,DOSSTI ;SET SINGLE INST MODE
2558 003760 105267 031427 INCB DEFSTP ;REMEMBER WE DEFAULTED TO INST STEP
2559 003764 000402 BR 25$
2560
2561 003766 005713 20$: TST (R3) ;TEST FOR SINGLE INST MODE
2562 003770 100024 BPL 40$ ;BR IF NOT SINGLE INSTRUCTION
2563 003772 004767 005126 25$: JSR PC,TSTRUN ;TEST FOR CPU RUNNING
2564 003776 103446 BCS 80$ ;BR IF CPU IS RUNNING
2565 004000 004767 005320 JSR PC,TSTVER ;CHECK FOR MICRO-VERSION MISMATCH
2566 004004 103443 BCS 80$ ;ABORT IF FATAL MISMATCH(C SET)
2567 004006 004767 177220 JSR PC,CONTSQ ;DO A STAR CPU CONTINUE
2568 ;CLR R1 ;(DONE BY 'PUSHU' RTN)
2569 004012 004767 003576 30$: JSR PC,TSTCLK ;TEST FOR CLOCK STOPPED
2570 004016 103436 RCS 80$ ;BR IF CLOCK STOPPED
2571 004020 004767 004116 JSR PC,TSTHAL ;TEST FOR CPU HALTED
2572 004024 103414 BCS 50$ ;BR IF HALTED
2573 004026 005201 INC R1 ;UPDATE TIMEOUT COUNTER
2574 004030 001370 BNE 30$ ;BR IF NOT TIMED-OUT YET
2575 004032 TYPEMES #TMEOUT,.CR ;TYPE TIMEOUT MESSAGE
2576 004040 000425 BR 80$ ;ABORT STEPPING
2577
2578 004042 004767 000364 40$: JSR PC,DOSTPG ;ENABLE PROGRAM I/O MODE
2579 004046 052714 000001 BIS #PROCD,(R4) ;ISSUE A PROCEED TO CPU CLOCK
2580 004052 004767 003334 JSR PC,TYPTIC ;TYPE CLOCK STATE
2581 004056 042713 000400 50$: BIC #SPCSYC,(R3) ;CLEAR SPACE-BAR SYNC FLAG
2582 004062 105767 031326 60$: TSTB ABORT ;TEST FOR COMMAND ABORTED
2583 004066 001012 BNE 80$ ;BR IF ABORTED(VIA CONTROL-C)
2584 004070 032713 001000 BIT #SPCSTP,(R3) ;TEST FOR SPACE-BAR STEP MODE
2585 004074 001404 BEQ 70$ ;BR IF NOT IN SPACE-BAR-STEP MODE
2586 004076 032713 000400 BIT #SPCSYC,(R3) ;WAIT FOR SPACE-SYNC TO SET
2587 004102 001767 BEQ 60$
2588 004104 000730 BR 20$ ;SPACE-SYNC SET. NOW DO NEXT STEP
2589
2590 004106 005367 031274 70$: DEC COUNT ;DECREASE STEP COUNTER
2591 004112 003325 BGT 20$ ;BR IF MORE STEPS TO DO
2592 004114 105767 031273 80$: TSTB DEFSTP ;SEE IF WE WERE DEFAULTING TO INST STEP
2593 004120 001402 BEQ 90$ ;BR IF WE WERE NOT DEFAULTING
2594 004122 004767 000266 JSR PC,DOSSTN ;RETURN CLOCK TO NORMAL MODE
2595 004126 042767 000020 031244 90$: BIC #INITLD,TCONTL ;MAKE SURE WE DO NOT AUTO-RESTART
2596 004134 000207 RTS PC
2597

```


ZZ-ESKAA-10.1 NEXT(PERFORM A STEP)
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 22-1
NEXT(PERFORM A STEP)

K 4

20-MAY-1986

Fiche 1 Frame K4

Sequence 49

2598

.DSABL LSB

```

2600                      .SBTTL  QUAD CLEAR
2601
2602                      .ENABL  LSB
2603
2604 004136              DOQCLE: ;PERFORM A QUAD CLEAR
2605                      ;R2-->TSTRUN
2606                      ;R4-->MCR
2607 004136 112767 000003 031255      MOVB  #QADLNH,CURLNH ;FORCE QUAD LENGTH
2608 004144 112767 000000 031250      MOVB  #PHYSPC,CURADS ;FORCE PHYSICAL ADDRESSING
2609 004152 004712                      JSR   PC,(R2) ;TEST FOR CPU RUNNING
2610 004154 103435                      BCS   20$ ;BR IF CPU RUNNING
2611 004156 042767 000007 032420      BIC   #7,EFFADR ;FORCE ADDRESS TO QUAD BOUNDARY
2612 004164 012700 000445 10$:      MOV   #CPREGE+1,R0 ;R0 GETS ADDRESS OF INT REG DEP RTN
2613 004170 004767 004536                      JSR   PC,PUSHU ;PUSH R0 ON MICRO-STACK
2614 004174 103425                      BCS   20$ ;BR IF PUSH FAILED
2615 004176 016700 032402      MOV   EFFADR,R0 ;R0 AND R1 GET ADDRESS TO QUAD CLEAR
2616 004202 016701 032400      MOV   EFFADR+2,R1
2617 004206 012702 000062      MOV   #T2,R2 ;R2 GETS ID BUS REG ADDRESS
2618 004212 004767 004540      JSR   PC,WRITID ;WRITE ADDRESS PARAMETER TO ID REG 'T2'
2619 004216 103417                      BCS   30$ ;BR IF ID WRITE FAILED
2620 004220 012737 000066 1/3020      MOV   #INTR36,a#T0IDLO ;PUT ADDRESS OF 'QUAD CLEAR' INT REG IN 'T0ID'
2621 004226 005037 173022      CLR   a#T0IDHI
2622 004232 052737 000200 173014      BIS   #RXDNE,a#RXDONE ;SET 'RX DONE'
2623 004240 052714 002000      BIS   #MAINTR,(R4) ;POP MICRO-STACK TO UPC,START INT REG DEPOSIT RTN
2624 004244 004767 177276      JSR   PC,COMWAT ;WAIT FOR STAR TO FINISH
2625 004250 004767 000004 20$:      JSR   PC,COMPAD ;UPDATE 'EFFADR'
2626 004254 103743                      BCS   10$ ;BR IF MORE ITERATIONS TO DO
2627 004256 000207 30$:      RTS   PC
2628
2629 004260              COMPAD: ;UPDATE 'EFFADR' AND CHECK FOR ITERATIONS
2630                      ;OUTPJTS: C BIT CLEAR IF COMMAND IS FINISHED
2631                      ; C BIT SET IF MORE ITERATIONS
2632 004260 004767 001026      JSR   PC,SETLAS ;'LASADD' GETS CONTENTS OF 'EFFADR'
2633 004264 032767 000400 031106      BIT   #MINSAD,TCONTL ;REVERSE ADDRESS UPDATE?
2634 004272 001403                      BEQ   40$ ;BR IF NO
2635 004274 004767 013276      JSR   PC,SETMNS ;'EFFADR' GETS 'EFFADR' MINUS DATALENGTH
2636 004300 000402                      BR    50$
2637
2638 004302 004767 013266 40$:      JSR   PC,SETPLS ;'EFFADR' GETS 'EFFADR' PLUS DATA LENGTH
2639 004306 105767 031102 50$:      TSTB  ABORT ;ABORT SET?
2640                      ;CLC
2641 004312 001003                      BNE   60$ ;BR TO EXIT IF YES
2642 004314 005367 031064      DEC   NEXTCT ;MORE TO DO?
2643 004320 002001                      BGE   55$ ;BRANCH IF YES
2644 004322 005727 60$:      TST   (PC)+ ;CLEAR C BIT
2645 004324 000261 55$:      SEC
2646 004326 000207      RTS   PC
2647
2648                      .DSABL  LSB

```

```

2650                      .SBTTL  SET STEP,CLOCK,SOMM
2651
2652                      .ENABL  LSB
2653
2654 004330      DOSSTI: ;SET STEP TO SINGLE INSTRUCTION
2655                      ;R3-->FLAG
2656                      ;R4-->MCR
2657 004330 052714 100000      BIS      #HLTREQ,(R4)      ;REQUEST STAR TO HALT
2658 004334 042713 040000      BIC      #IGNORE,(R3)      ;ALLOW CLOCK STOPS TO REPORT
2659 004340 052713 100000      BIS      #SNGINS,(R3)      ;REMEMBER SINGLE INSTRUCTION STEP MODE
2660 004344 042714 000006      STRTCK: BIC      #STS!SBC,(R4) ;CLEAR SINGLE BUS CYCLE AND TIME STATE
2661 004350 052714 000001      BIS      #PROCED,(R4)      ;START CLOCK
2662 004354 000207      RTS      PC
2663
2664
2665 004356      DOSSTB: ;SET CLOCK TO SINGLE BUS CYCLE
2666                      ;R4-->MCR
2667 004356 042714 000004      BIC      #STS,(R4)
2668 004362 052714 000002      BIS      #SBC,(R4)      ;SET SINGLE BUS CYCLE CLOCK BIT
2669 004366 042767 100000 031226 BIC      #SNGINS,FLAG      ;CLEAR SINGLE INSTRUCTION MODE
2670 004374 000207      RTS      PC
2671
2672 004376      DOSSTS: ;SET CLOCK TO SINGLE TIME STATE
2673                      ;R4-->MCR
2674 004376 004767 177754      JSR      PC,DOSSTB      ;SET SINGLE BUS CYCLE FIRST
2675 004402 042714 000002      BIC      #SBC,(R4)      ;CLEAR SBC
2676 004406 052714 000004      BIS      #STS,(R4)      ;SET SINGLE TIME STATE
2677 004412 000207      RTS      PC
2678
2679 004414      DOSSTN: ;SET CLOCK TO FREE RUN
2680                      ;R3-->'FLAG'
2681 004414 042713 140000      BIC      #SNGINS!IGNORE,(R3) ;CLEAR SNG INST STEP, ALLOW CLOCK REPORTING
2682 004420 000751      BR      STRTCK
2683
2684
2685 004422      DOSTER: ;SET TERMINAL FILL
2686 004422 116767 032134 031117 MOVB      DATATO,TERFIL
2687 004430 000207      RTS      PC
2688
2689
2690 004432      DOSTPG: ;SET PROGRAM I/O MODE
2691 004432 105267 031157      INCB      PGMION
2692 004436 042767 100400 031066 BIC      #ROFLAG!PRNINH,TCTFLG ;CLEAR PRINT-INHIBIT, RUBOUT SERVICE
2693 004444 052737 000200 173016 SETTXR: BIS      #TXRDY,#TXREAD ;SET 'TX READY'
2694 004452 000207      RTS      PC
2695
2696
2697 004454      DOSTSO: ;SET SOMM ON CIB
2698                      ;R4-->MCR
2699 004454 052714 000100      BIS      #SOMMB,(R4)
2700 004460 000207      RTS      PC
2701
2702
2703 004462      DOSTCF: ;SET CLOCK FREQ TO FAST
2704                      ;R4-->MCR

```

```
2705 004462 004767 000006      JSR    PC,DOSTCN      ;SET FREQ TO NORMAL FIRST
2706 004466 052714 000010      BIS    #FREQ0,(R4)
2707 004472 000207      RTS    PC
2708
2709
2710 004474      DOSTCN: ;SET CLOCK FREQ TO NORMAL
2711      ;R4-->MCR
2712 004474 042714 000030      BIC    #FREQ0!FREQ1,(R4)
2713 004500 000207      RTS    PC
2714
2715
2716 004502      DOSTCS: ;SET CLOCK FREQ TO SLOW
2717      ;R4-->MCR
2718 004502 004767 177766      JSR    PC,DOSTCN      ;SET CLOCK TO NORMAL FREQ FIRST
2719 004506 052714 000020      BIS    #FREQ1,(R4)
2720 004512 000207      RTS    PC
2721
2722      .DSABL  LSB
```

```

2724                      .SBTTL  EXAMINE,DEPOSIT
2725
2726                      .ENABL  LSB
2727
2728 004514      DODEEX:  ;PERFORM A DEPOSIT OR EXAMINE
2729                      ;CALLED BY 'EXECUT'
2730                      ;      'DEEXBY'=0 IF EXAMINE
2731                      ;      'DEEXBY'=1 IF DEPOSIT
2732
2733 004514 012701 036604'  MOV      #EFFADR,R1      ;R1 GETS USEFUL POINTER
2734 004520 012146          MOV      (R1)+,-(SP)      ;SAVE EFFECTIVE ADDRESS ON STACK
2735 004522 011146          MOV      (R1)+,-(SP)
2736 004524 126727 030672 000001  CMPB    CURADS,#VIRSPC  ;SEE IF RELOCATION TO BE APPLIED
2737 004532 003006          BGT      10$              ;BR IF NOT VIRT OR PHYS ADDRESS
2738 004534 005741          TST      -(R1)            ;POINT R1 TO EFFADR AGAIN
2739 004536 066721 015760  ADD      RELOCA,(R1)+      ;ADD RELOCATION REGISTER TO EFFECTIVE ADDRESS
2740 004542 005511          ADC      (R1)
2741 004544 066711 015754  ADD      RELOCA+2,(R1)
2742 004550 004767 000246 10$:  JSR      PC,DEEXPM      ;DO THE DEPOSIT EXAMINE PRIMITIVE
2743 004554 103475          BCS      50$              ;BR IF ERROR ON DE/EX
2744 004556 105767 030630  TSTB    DEEXBY            ;TEST FOR EXAMINE
2745 004562 001072          BNE      50$              ;BR IF DEPOSIT(SKIP REPORTING)
2746 004564 016767 031752 015674  MOV     DATAFR,LASDAT      ;SAVE 'LAST DATA'
2747 004572 016767 031746 015670  MOV     DATAFR+2,LASDAT+2
2748 004600 116700 030616          MOVB   CURADS,R0      ;R0 GETS CODE FOR CURRENT ADDRESS SPACE
2749 004604 020027 000004          CMP     R0,#IDBSPC      ;CHECK FOR ID BUS REF(MAKE A CHECK FOR PSL)
2750 004610 001010          BNE      15$              ;BR IF NOT ID BUS
2751 004612 026727 031766 000017  CMP     EFFADR,#17      ;CHECK FOR PSL'S ADDRESS
2752 004620 001004          BNE      15$              ;BR IF NOT PSL REFERENCE
2753 004622          TYPEMES #PSLSTR,,CR      ;TYPE SPACES IN LIEU OF ADDRESS
2754 004630 000431          BR        45$
2755
2756 004632 006300 15$:  ASL      R0
2757 004634          TYPEMES IDNTTB(R0),,CR      ;TYPE IDENTIFIER STRING
2758 004642 020027 000002          CMP     R0,#VIRSPC*2      ;CHECK FOR A VIRTUAL REFERENCE
2759 004646 001010          BNE      20$              ;BR IF NOT VIRTUAL REFERENCE
2760 004650 012703 036610'  MOV      #GOTID,R3      ;READ TRANSLATED ADDRESS FROM ID REG 'I3'
2761 004654 010346          MOV      R3,-(SP)          ;STACK R3 FOR USE IN NEXT STEP BELOW
2762 004656 012702 000063          MOV      #T3,R2      ;R2 GETS ADDRESS OF ID REG 'T3'
2763 004662 004767 004154          JSR      PC,READID      ;READ ID BUS REG
2764 004666 000402          BR        30$
2765
2766 004670 012746 036604' 20$:  MOV      #EFFADR,-(SP)      ;STACK POINTER TO ADDRESS
2767 004674 012746 000004 30$:  MOV      #4,-(SP)      ;STACK LENGTH OF ADDRESS IN BYTES
2768 004700 004767 000100          JSR      PC,R2GRAD      ;R2 GETS CURRENT RADIX VALUE
2769 004704 010246          MOV      R2,-(SP)          ;STACK R2 FOR CONVERTER
2770 004706 004767 140022'  JSR      PC,CONVRT      ;CONVERT ADDRESS TO ASCII STRING
2771 004712          TYPEMES          ;TYPE ADDRESS STRING
2772 004714          TYPEMES #TWOSPC      ;TYPE 2 SPACES
2773 004722 012746 036542'  MOV      #DATAFR,-(SP)      ;STACK POINTER TO RETURNED DATA
2774 004726 016746 015564          MOV      LNHDAT,-(SP)      ;STACK LENGTH OF DATA IN BYTES
2775 004732 004767 000046          JSR      PC,R2GRAD      ;R2 GETS CURRENT RADIX VALUE
2776 004736 010246          MOV      R2,-(SP)          ;STACK R2 FOR CONVERTER
2777 004740 004767 140022'  JSR      PC,CONVRT      ;CONVERT RETURNED DATA TO ASCII STRING
2778 004744          TYPEMES          ;TYPE RETURNED DATA STRING

```

```
2779 004746 000406          BR      60$
2780
2781 004750 016767 031606 015510 50$:  MOV    DATATO,LASDAT      ;SAVE 'LAST DATA'
2782 004756 016767 031602 015504      MOV    DATATO+2,LASDAT+2
2783 004764 012667 031616          60$:  MOV    (SP)+,EFFADR+2  ;RESTORE EFFECTIVE ADDRESS
2784 004770 012667 031610          MOV    (SP)+,EFFADR
2785 004774 004767 177260          JSR     PC,COMPAD      ;UPDATE 'EFFADR', TEST FOR ITERATIONS
2786 005000 103645          BCS     DODEEX      ;BR IF MORE ITERATIONS
2787 005002 000207          90$:  RTS     PC
2788
2789 005004          R2GRAD: ;R2 <-- 16 IF RADIX CURRENTLY HEX
2790                      ;R2 <-- 8 IF RADIX CURRENTLY NOT HEX
2791 005004 012702 000010          MOV     #8,R2      ;ASSUME OCTAL
2792 005010 105767 030404          TSTB    CURRAD      ;CURRENT RADIX HEX?
2793 005014 001001          BNE     100$      ;BR IF NOT
2794 005016 006302          ASL      R2      ;CHANGE THE 8 TO A 16
2795 005020 000207          100$:  RTS     PC
2796
2797                      .DSABL  LSB
```

```

2799 .ENABL LSB
2800
2801 005022 DEEXPM: ;DEPOSIT OR EXAMINE SOMETHING
2802 ; 'DEEXBY'=0 IF EXAMINE, 1 IF DEPOSIT
2803 ; 'EFFADR'=ADDRESS TO USE
2804 ; 'DATATO'=DATA FOR DEPOSIT
2805 ; 'CURADS'=CODE FOR ADDRESS SPACE TO USE
2806 ; 0=PHYS,1=VIRT,2=GEN,3=INTERNAL,4=IDBUS,
2807 ; 5=CONSOLE,6=VBUS
2808 ; 'CURLNH'=CODE FOR DATA LENGTH
2809 ; 0=BYTE,1=WORD,2=LONG,3=QUAD
2810 ;
2811 ;OUTPUTS: C BIT SET IF ERROR, ELSE
2812 ; 'DATAFR'=EXAMINED DATA
2813
2814 005022 004077 140054' JSR R0,@RSAPV ;SAVE R0-R5,POINT R3 TO 'FLAG'
2815 005026 042713 000011 BIC #SECHLF!QADTYP,(R3) ;CLEAR SOME FLAGS
2816 005032 105067 030561 CLR B ;CLEAR TIMEOUT FLAG
2817 005036 004767 000274 JSR PC,SETLNH ;'LNHDAT'<--LENGTH OF DATA IN BYTES,R2<--DATA LNH CODE
2818 005042 020227 000003 CMP R2,#QADLNH ;CHECK FOR QUAD LENGTH
2819 005046 002403 BLT 10$ ;BR IF NOT QUAD
2820 005050 005302 DEC R2 ;CHANGE TO LONG
2821 005052 052713 000010 BIS #QADTYP,(R3) ;REMEMBER QUAD LENGTH
2822 005056 010267 015436 10$: MOV R2,LNHCD ;SAVE DATA LENGTH FOR MICRO-CODE
2823 005062 116702 030334 20$: MOV B CURADS,R2 ;R2 GETS ADDRESS SPACE CODE
2824 005066 006302 ASL R2
2825 005070 004772 005232' JSR PC,@EXDEV(C(R2) ;DO 7 WAY BRANCH ON ADDRESS SPACE
2826 005074 103454 BCS 50$ ;BR IF FAILURE ON EX OR DE
2827 005076 012703 035622' MOV #FLAG,R3 ;USEFUL POINTER TO R3
2828 005102 126727 030314 000002 CMP B CURADS,#GENSPC ;CHECK FOR GEN REG SPACE
2829 005110 002042 BGE 40$ ;SKIP QUAD TEST FOR ALL EXCEPT PHYS AND VIRT
2830 005112 012701 036604' MOV #EFFADR,R1 ;R1 GETS POINTER TO EFFADR
2831 005116 032713 000010 BIT #QADTYP,(R3) ;TEST FOR QUAD LENGTH
2832 005122 001435 BEQ 40$ ;BR IF NOT QUAD
2833 005124 032713 000001 BIT #SECHLF,(R3) ;CHECK FOR SECOND PART OF QUAD DONE
2834 005130 001023 BNE 30$ ;BR IF SECOND HALF DONE
2835 005132 052713 000001 BIS #SECHLF,(R3) ;REMEMBER SECOND HALF BEING DONE
2836 005136 012146 MOV (R1)+,-(SP) ;SAVE EFFADR
2837 005140 011146 MOV (R1)+,-(SP)
2838 005142 016746 031414 MOV DATATO,-(SP) ;SAVE FIRST TWO WORDS OF QWORD DEPOSIT
2839 005146 016746 031412 MOV DATATO+2,-(SP) ;
2840 005152 062741 000004 ADD #4,-(R1) ;ADD 4 TO ADDRESS
2841 005156 005561 000002 ADC 2(R1)
2842 005162 016767 031400 031372 MOV DATATO+4,DATATO ;SET DATA FOR SECOND DEPOSIT
2843 005170 016767 031374 031366 MOV DATATO+6,DATATO+2
2844 005176 000731 BR 20$
2845
2846 005200 012667 031360 30$: MOV (SP)+,DATATO+2 ;RESTOR FIRST TWO WORDS OF QWORD DEPOSIT
2847 005204 012667 031352 MOV (SP)+,DATATO ;
2848 005210 012661 000002 MOV (SP)+,2(R1) ;RESTORE EFFADR
2849 005214 012611 MOV (SP)+,(R1)
2850 005216 105767 030375 40$: TST B TIMEOUT ;TIMEOUT OR ERROR?
2851 ;CLC
2852 005222 001401 BEQ 50$ ;BR IF NOT
2853 005224 000261 SEC

```

ZZ-ESKAA-10.1 EXAMINE,DEPOSIT
VIO-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 26-1
EXAMINE,DEPOSIT

E 5

20-MAY-1986

Fiche 1 Frame E5

Sequence 56

2854 005226 000167 002672 50\$: JMP REPLAC ;RESTORE R0-R5,THEN RETURN


```

2856 005232 005360'      EXDEV: .WORD  PHEXDE      ;MICRO-ASSISTED ROUTINE
2857 005234 005364'      .WORD  VIEXDE      ;
2858 005236 005420'      .WORD  GEEXDE      ;
2859 005240 005434'      .WORD  INEXDE      ;
2860 005242 005652'      .WORD  IDEXDE      ;NON-MICRO-ASSISTED
2861 005244 006176'      .WORD  COEXDE      ;
2862 005246 006050'      .WORD  VBEXDE      ;
2863
2864 005250 021537' 021537' 021544' IDNTTB: .WORD  PHYIDN,PHYIDN,GENIDN,INTIDN,IDBIDN,CONIDN,VBUIDN
      005256 021551' 021556' 021563'
      005264 021570'
2865
2866 005266 022516' 022516' 005304' ADUPTB: .WORD  LNHDAT,LNHDAT,NUMB1,NUMB1,NUMB1,LNHDAT,NUMB1
      005274 005304' 005304' 022516'
      005302 005304'
2867 005304 000001      NUMB1: .WORD  1
2868 005306 001      002      004  DATLTB: .BYTE  1,2,4,8.      ;NUMBER OF BYTES IN BYTE,WORD,LONG QUAD
      005311 010
      .EVEN
2869
2870
2871 005312 016767 031266 031252 SETLAS: MOV  EFFADk,LASADD
2872 005320 016767 031262 031246      MOV  EFFADR+2,LASADD+2
2873 005326 116767 030070 015152      MOVB  CURADS,LASADS      ;REMEMBER ADDRESS SPACE
2874 005334 000207      RTS  PC
2875
2876 005336      SETLNH: ;'LNHDAT' GETS DATA LENGTH IN BYTES
2877      ;R2 GETS DATA LENGTH CODE
2878      ;NOTE: THIS ROUTINE MUST NOT CHANGE THE C BIT
2879 005336 116702 030057      MOVB  CURLNH,R2      ;R2 GETS DATA LENGTH CODE
2880 005342 100002      BPL  60$      ;BR IF CURRENT LENGTH IS VALID
2881 005344 116702 030054      MOVB  DEFLNH,R2      ;SUBSTITUTE DEFAULT LENGTH
2882 005350 116267 005306' 015140 60$: MOVB  DATLTB(R2),LNHDAT ;SET 'LNHDAT' TO LENGTH OF DATA IN BYTES
2883 005356 000207      RTS  PC
2884
2885      .DSABL  LSB

```

```

2887 .SBTTL MICRO-ASSISTED EXAMINE/DEPOSIT ROUTINES
2888
2889 .ENABL LSB
2890
2891 ; **** NOTE -- EDIT-20 MADE MAJOR CHANGES HERE
2892 005360 005002 PHEXDE: CLR R2 ;SET UP TO INDICATE 'PHYSICAL'
2893 005362 000402 BR 5$
2894 005364 012702 000001 VIEXDE: MOV #1,R2 ;SET UP TO INDICATE 'VIRTUAL'
2895 005370 004767 003530 5$: JSR PC,TSTRUN ;CHECK FOR CPU RUNNING
2896 005374 103001 BCC 6$ ;BR IF NOT
2897 005376 000207 RTS PC
2898 005400 006002 6$: ROR R2 ;SET C-BIT TO INDICATE PHYSICAL/VIRTUAL
2899 ; **** END OF EDIT-20
2900 005402 004767 001730 JSR PC,STCLMP ;MEMORY MAPPING ENABLE GETS C BIT
2901 005406 004767 003204 JSR PC,TSTTY2 ;CLEAR CODE 2 MICRO-ERRORS
2902 005412 004067 000024 LOADDE: JSR R0,MICAST ;CONTINUE IN COMMON MICRO-ASSISTED RTN
2903 005416 000440 .WORD CPHYSE ;MICRO-ADDRESS OF RTN TO USE
2904
2905 005420 042767 177760 031156 GEEXDE: BIC #177760,EFFADR ;CLEAR UNNEEDED ADDRESS BITS
2906 005426 004067 000010 JSR R0,MICAST ;CONTINUE IN COMMON MICRO-ASSISTED RTN
2907 005432 000442 .WORD CGREGE ;MICRO-ADDRESS OF RTN TO USE
2908
2909 005434 004067 000002 INEXDE: JSR R0,MICAST ;CONTINUE IN COMMON MICRO-ASSISTED RTN
2910 005440 000444 .WORD CPREGE ;MICRO-ADDRESS OF RTN TO USE
2911
2912
2913 005442 MICAST: ;ROUTINE TO PERFORM A MICRO-ASSISTED EXAMINE OR DEPOSIT
2914 ;CALLED BY JSR R0,MICAST
2915 ; MICRO-ROUTINE ADDRESS TRAILS THE CALL
2916 ; RETURN IS MADE TO 'DEEXPM' RTN(RTN ADDRESS AT 2(SP))
2917 005442 012000 MOV (R0)+,R0 ;R0 GETS MICRO-RTN ADDRESS
2918 005444 005726 TST (SP)+ ;REMOVE SAVED R0 FROM STACK
2919 005446 004767 003452 JSR PC,TSTRUN ;TEST FOR STAR CPU RUNNING
2920 005452 103476 BCS 50$ ;BR IF STAR IS RUNNING
2921 005454 105767 027732 TSTB DEEXBY ;TEST FOR EX OR DE
2922 005460 001401 BEQ 10$ ;BR IF EXAMINE
2923 005462 005200 INC R0 ;ADD 1 TO GET ADDRESS OF DEPOSIT RTN
2924 005464 004767 003242 10$: JSR PC,PUSHU ;PUSH R0 ON MICRO-STACK
2925 005470 103467 BCS 50$ ;BR IF CLOCK STOPPED
2926 005472 016700 015022 MOV LNHCD,R0 ;R0 GETS CODE FOR DATA LENGTH
2927 005476 012702 000061 MOV #T1,R2 ;R2 GETS ADDRESS OF ID REG T1
2928 005502 004767 003250 JSR PC,WRITID ;WRITE LENGTH CODE TO ID REG T1
2929 005506 103460 BCS 50$ ;BR IF ID BUS WRITE FAILED
2930 005510 105767 027676 TSTB DEEXBY ;TEST FOR EX OR DE
2931 005514 001411 BEQ 20$ ;BR IF EXAMINE
2932 005516 016700 031040 MOV DATATO,R0 ;R0 AND R1 GET DATA TO DEPOSIT
2933 005522 016701 031036 MOV DATATO+2,R1
2934 005526 012702 000062 MOV #T2,R2 ;R2 GETS ADDRESS OF ID REG T2
2935 005532 004767 003220 JSR PC,WRITID ;WRITE DEPOSIT DATA TO ID REG T2
2936 005536 103444 BCS 50$ ;BR IF ID BUS WRITE FAILED
2937 005540 012700 173020 20$: MOV #T0IDLO,R0 ;POINT R0 TO 'T0ID' REG
2938 005544 016720 031034 MOV EFFADR,(R0)+ ;PUT ADDRESS INTO 'T0ID' REG
2939 005550 016720 031032 MOV EFFADR+2,(R0)+
2940 ;NOTE:R0 NOW POINTS TO 'FMIDLO'
2941 005554 052737 000200 173014 BIS #RXDONE,a#RXDONE ;SET RXDONE

```

2942	005562	052714	002000		BIS	#MAINTR,(R4)	;POP MICRO-STACK TO UPC
2943	005566	004767	002126		JSR	PC,CWAIT	;WAIT FOR FUNCTION TO COMPLETE
2944	005572	103426			BCS	50\$;BR IF TIMEOUT ON WAIT
2945	005574	105767	027612		TSTB	DEEXBY	;TEST FOR EX OR DE
2946	005600	001011			BNE	40\$;BR IF DEPOSIT
2947	005602	012702	036542		MOV	#DATAFR,R2	;POINT R2 TO RETURNED DATA AREA
2948	005606	032767	000001	030006	BIT	#SECHLF,FLAG	;TEST FOR SECOND HALF OF QUAD EXAMINE
2949	005614	001401			BEQ	30\$;BR IF NOT SECOND HALF OF QUAD
2950	005616	022222			CMP	(R2)+,(R2)+	;POINT R2 TO DATAFR+4
2951	005620	012022		30\$:	MOV	(R0)+,(R2)+	;SAVE RETURNED DATA
2952	005622	012022			MOV	(R0)+,(R2)+	
2953	005624	105767	014664	40\$:	TSTB	LODFLG	;HERE FROM 'LOAD A FILE'? (EDIT-21)
2954	005630	001005			BNE	45\$;IF SO, DON'T RESTORE MME YET
2955	005632	105767	031123		TSTB	XLOFLG	; DOING X LOAD?
2956	005636	001094			BNE	50\$; BRANCH IF YES
2957	005640	004767	001536		JSR	PC,RESTMM	;RESTORE 'MME'(MEMORY MAPPING ENABLE)
2958	005644	004767	002106	45\$:	JSR	PC,TSTERR	;TEST FOR SUCCESS ON FUNCTION
2959	005650	000207		50\$:	RTS	PC	; (C BIT SET BY TSTERR IF NOT SUCCESSFUL)
2960							
2961					.DSABL	LSB	

```

2963          .SBTTL    EXAMINE ID BUS
2964
2965          .ENABL    LSB
2966
2967 005652          IDXDDE: ;EXAMINE OR DEPOSIT TO ID BUS SPACE
2968 005652 012700 036604'  MOV    #EFFADR,R0      ;POINT R0 TO ADDRESS
2969 005656 042710 177700  BIC    #177700,(R0)    ;TRUNCATE ADDRESS
2970 005662 012002          MOV    (R0)+,R2      ;R2 GETS ADDRESS
2971 005664 005010          CLR    (R0)          ;CLEAR ADDRESS UPPER BITS
2972 005666 012703 036542'  MOV    #DATAFR,R3      ;POINT R3 TO UOUPUT DATA AREA
2973 005672 105767 027514  TSTB   DEEXBY        ;TEST FOR EXAMINE
2974 005676 001026          BNE    20$           ;BR IF NOT EXAMINE
2975 005700 032737 000040 173032  BIT    #CLKSTD,a#MCR    ;TEST FOR CLOCK STOPPED
2976 005706 001414          BEQ    10$           ;BR IF CLOCK RUNNING
2977 005710 012700 173030  MOV    #IDCNTRL,R0      ;DO A STATIC ID BUS EXAMINE
2978 005714 010210          MOV    R2,(R0)        ;ADDRESS TO IDCNTL REG
2979 005716 052710 000200  BIS    #IDMANT,(R0)    ;SET 'ID MAINTENANCE' BIT
2980 005722 013723 173006  MOV    a#IDDATL,(R3)+ ;GET ID DATA
2981 005726 013713 173010  MOV    a#IDDATA,(R3)
2982 005732 042710 000200  BIC    #IDMANT,(R0)    ;CLEAR 'ID MAINTENANCE' BIT
2983 005736 000417          BR     30$
2984
2985 005740 004767 003160  10$: JSR    PC,TSTRUN      ;TEST FOR STAR RUNNING
2986 005744 103414          BCS    30$           ;BR IF STAR IS RUNNING
2987 005746 004767 003070  JSR    PC,READID      ;READ ID BUS
2988 005752 000411          BR     30$
2989
2990 005754 004767 003144  20$: JSR    PC,TSTRUN      ;TEST FOR STAR RUNNING
2991 005760 103406          BCS    30$           ;BR IF RUNNING
2992 005762 016700 030574  MOV    DATATO,R0      ;R0 ,R1 GET DATA TO WRITE
2993 005766 016701 030572  MOV    DATA^0+2,R1
2994 005772 004767 002760  JSR    PC,WRITID      ;WRITE R0,R1 TO ID AS ADDRESSED BY R2
2995 005776 000207          30$: RTS    PC
2996
2997          .DSABL    LSB

```

ZZ-ESKAA-10.1 EXAMINE/DEPOSIT STAR PC
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 30
 EXAMINE/DEPOSIT STAR PC

2999				.SBTTL	EXAMINE/DEPOSIT STAR PC
3000					
3001	006000			EXDEPC:	;ROUTINE TO EXAMINE OR DEPOSIT TO STAR PC
3002				;INPUTS:	IF<'DEEXBY'=0> THEN <EXAMINE PC> ELSE <DEPOSIT>
3003				:	(CONTENTS OF 'DATATO' ARE DEPOSIT DATA)
3004				;OUTPUTS:	'DATAFR' CONTAINS PC CONTENTS IF EXAMINE
3005	006000	012700	036604'	MOV	#EFFADR,R0
3006	006004	012046		MOV	(R0)+,-(SP) ;SAVE EFFADR
3007	006006	012046		MOV	(R0)+,-(SP)
3008	006010	005040		CLR	-(R0) ;CLEAR UPPER WORD OF 'EFFADR'
3009	006012	012740	000017	MOV	#17,-(R0) ;SET ADDRESS TO 17(PC'S ADDRESS OF COURSE)
3010	006016	112767	000002 027376	MOVB	#GENSPC,CURADS ;SET ADDRESS SPACE TO GEN REG SPACE
3011	006024	112767	000002 027367	MOVB	#LNLNH,CURLNH ;SET DATA LENGTH TO LONG WORD
3012	006032	004767	176764	JSR	PC,DEEXPM ;DO THE DEPOSIT EXAMINE PRIMITIVE
3013	006036	012667	030544	MOV	(SP)+,EFFADR+2
3014	006042	012667	030536	MOV	(SP)+,EFFADR ;RESTORE ADDRESS
3015	006046	000207		RTS	PC

```

3017 .SBTTL VBUS EXAMINE
3018
3019 .ENABL LSB
3020
3021 006050 VBEXDE: ;EXAMINE VBUS(NO DEPOSIT TO VBUS)
3022 ; EFFADR=ADDRESS TO EXAMINE
3023 ;OUTPUTS: DATAFR HOLDS 16 BYTES OF VBUS CHANNEL DATA
3024 ; 'LNHDAT' HOLDS # OF BYTES EXAMINED
3025
3026 006050 105767 027336 TSTB DEEXBY ;TEST FOR DEPOSIT
3027 006054 001043 40$ ;BR IF DEPOSIT
3028 006056 016702 030522 EXUPC: MOV EFFADR,R2 ;R2 GETS ADDRESS TO EXAMINE
3029 006062 042702 177770 BIC #177770,R2 ;TRUNCATE ADDRESS TO 3 BITS
3030 006066 005003 CLR R3
3031 006070 156203 006166' BISB MASKS(R2),R3 ;R3 GETS CHANNEL MASK
3032 006074 000303 SWAB R3 ;MASK GOES TO UPPER BYTE OF R3
3033 006076 012702 000020 MOV #20,R2 ;R2 GETS LENGTH OF LARGEST CHANNEL(BYTES)
3034 006102 012700 036542' MOV #DATAFR,R0 ;POINT R0 TO OUTPUT AREA
3035 006106 010267 014404 MOV R2,LNH DAT ;SAVE LENGTH OF CHANNEL FOR REPORTER
3036 006112 012704 173036 MOV #VBUSR,R4 ;POINT R4 TO VBUS CONTROL REGISTER
3037 006116 052714 000002 BIS #VLOAD,(R4) ;LOAD THE VBUS FLOPS
3038 006122 042714 000002 BIC #VLOAD,(R4) ;DEASSERT THE LOAD SIGNAL
3039 006126 012705 000010 10$: MOV #8.,R5 ;SET R5 TO COUNT FOR 1 BYTE
3040 006132 005001 CLR R1
3041 006134 20$:: CLC ;CLC NOT NEEDED SINCE 'CLR' CLEARS
3042 ;AND 'ROR' CLEARS
3043 006134 030314 BIT R3,(R4) ;TEST FOR A ONE IN THIS BIT OF CHANNEL
3044 006136 001401 BEQ 30$ ;BR IF THIS BIT IS A ZERO
3045 006140 000261 SEC
3046 006142 006001 30$: ROR R1 ;R1 GETS C BIT SHIFTED IN
3047 006144 052714 000001 BIS #VCLK,(R4) ;SHIFT THE VBUS ONE PLACE
3048 006150 005305 DEC R5 ;REDUCE SHIFT COUNT BY ONE
3049 006152 003370 BGT 20$ ;BR IF ONE BYTE NOT SHIFTED YET
3050 006154 000301 SWAB R1 ;MOVE DATA BYTE TO LOWER BYTE OF R1
3051 ;CLC
3052 006156 110120 MOVB R1,(R0)+ ;SAVE THE BYTE
3053 006160 005302 DEC R2 ;REDUCE CHANNEL LENGTH COUNTER BY ONE
3054 006162 003361 BGT 10$ ;BR IF MORE BYTES TO GET
3055 006164 000207 40$: RTS PC
3056
3057 006166 001 002 004 MASKS: .BYTE 1,2,4,10,20,40,100,200 ;VBUS CHANNEL MASKS
3058 006171 010 020 040
3059 006174 100 200
3058
3059 .EVEN ;JUST IN CASE
3060 .DSABL LSB

```

```

3062                                .ENABL  LSB
3063
3064 006176          COEXDE: ;EXAMINE OR DEPOSIT TO CONSOLE'S OWN ADDRESS SPACE
3065                                ;NOTE: ALL XFERS ARE DONE ONE BYTE AT A TIME TO AVOID ODD
3066                                ; ADDRESS TRAPS
3067                                ;
3068                                ; COMPLETION CODE:
3069                                ; C BIT CLEAR IF EX/DE OK
3070                                ; C BIT SET IF NON EXISTANT MEMORY
3071                                ;
3072 006176 016701 014314          MOV     LNHDAT,R1      ;R1 GETS LENGTH OF XFER IN BYTES
3073 006202 012702 036542'        MOV     #DATAFR,R2    ;ASSUME EXAMINE. POINT R2 TO DATA OUTPUT AREA
3074 006206 016703 030372        MOV     EFFADR,R3      ;R3 GETS ADDRESS TO EXAMINE
3075 006212 022703 040000        CMP     #MEMSIZ,R3     ;ADDRESS IN RANGE?
3076 006216 101412                BLOS    20$           ;BRANCH IF NO
3077 006220 105767 027166        TSTB    DEEXBY        ;CHECK FOR EX OR DE
3078 006224 001403                BEQ     10$           ;BR IF EXAMINE
3079 006226 010302                MOV     R3,R2          ;REARRANGE POINTER FOR DEPOSIT
3080 006230 012703 036562'        MOV     #DATATO,R3    ;R3 POINTS TO DATA TO DEPOSIT
3081 006234 112322          10$:  MOVB    (R3)+,(R2)+    ;XFER A BYTE EITHER WAY
3082 006236 005301                DEC     R1            ;REDUCE XFER COUNTER BY ONE
3083 006240 003375                BGT     10$           ;BR IF MORE TO XFER
3084 006242 005727                TST     (PC)+         ;CLEAR C BIT
3085 006244 000261          20$:  SEC                     ;ERROR RETURN
3086 006246 000207                RTS     PC
3087                                .DSABL  LSB

```

```

3089 .SBTTL EXAMINE INSTRUCTION REGISTER(IR)
3090
3091 .ENABL LSB
3092
3093 006250 DOIR: ;READ OP-CODE, SPECIFIER, AND EXECUTION POINT COUNTER
3094 ;AND DISPLAY IN THAT ORDER
3095 ;VBUS CHANNEL 3 IS READ AND THE APPROPRIATE FIELDS ARE
3096 ;EXTRACTED AND DISPLAYED. THIS RTN WORKS WITH CLOCK ON OR OFF.
3097 ;NOTE THAT OP-CODE AND SPECIFIER ARE 'BACKWARDS' WHEN READ
3098 ;AND THE BITS MUST BE REVERSED BEFORE DISPLAYING
3099 ;INPUTS: R1=0
3100 ;OUTPUTS: NONE
3101
3102 006250 TYPEMES #IRIDN,,CR ;TYPE IDENT STRING
3103 006256 012767 000003 030320 MOV #3,EFFADR ;SET ADDRESS TO 3
3104 006264 004767 177566 JSR PC,EXUPC ;READ VBUS CHANNEL 3 TO 'DATAFR'
3105 006270 116700 030252 MOV DATAFR+4,R0 ;GET OP-CODE BYTE
3106 006274 004767 000052 JSR PC,20$ ;REVERSE BITS IN LOWER BYTE OF R0 AND PRINT
3107 006300 116700 030243 MOV DATAFR+5,R0 ;GET SPECIFIER BYTE
3108 006304 004767 000042 JSR PC,20$ ;REVERSE BYTE THEN PRINT
3109 006310 116700 030241 MOV DATAFR+11,,R0 ;GET EXECUTION POINT COUNTER
3110 006314 006200 ASR R0 ;3 BITS WE WANT ARE 4 AWAY FROM RIGHT END
3111 006316 006200 ASR R0
3112 006320 006200 ASR R0
3113 006322 006200 ASR R0
3114 006324 042700 177770 BIC #177770,R0 ;CLEAR ALL EXCEPT 3 BITS WE WANT
3115 006330 110067 014212 10$: MOV R0,FILENM ;SAVE THE BYTE TO PRINT
3116 006334 012746 022546' MOV #FILENM,-(SP) ;STACK ADDRESS OF BYTE
3117 006340 012746 000001 MOV #1,-(SP) ;STACK LENGTH OF DATA
3118 006344 004767 002066 JSR PC,CNTYPC ;CONVERT BYTE AND PRINT
3119 006350 000207 RTS PC
3120
3121 006352 005001 20$: CLR R1 ;REVERSE AND PRINT THE BYTE IN R0
3122 006354 012746 000010 MOV #8,-(SP) ;COUNT FOR 8 SHIFTS
3123 006360 006200 30$: ASR R0 ;BIT OF R0 TO C BIT
3124 006362 006101 ROL R1 ;R1 GETS THE C BIT
3125 006364 005316 DEC (SP) ;BUMP COUNT
3126 006366 003374 BGT 30$ ;BR IF MORE SHIFTS
3127 006370 005726 TST (SP)+ ;CLEAR STACK
3128 006372 010100 MOV R1,R0 ;GET REVERSED BYTE TO R0
3129 006374 004767 177730 JSR PC,10$ ;PRINT BYTE IN R0
3130 006400 TYPEMES #TWOSPC ;TYPE 2 SPACES
3131 006406 000207 RTS PC
3132
3133 .DSABL LSB
3134 006410 105067 027557 DOENDX: CLRB NODRV1 ;CLEAR THE 'NO DRIVE 1' FLAG
3135 006414 000207 RTS PC

```



```

3137          .SBTTL  SHOW CONSOLE STATE
3138
3139          .ENABL  LSB
3140
3141 006416      DOSHOW: ;DISPLAY DEFAULTS,CPU STATE,STEP MODE,CLOCK MODE, FILL
3142                ;R3-->'FLAG'
3143                ;R4-->MCR
3144 006416      TYPEMES #CPUIS,,CR          ;TYPE 'CPU '
3145 006424      MOV      #RUNNIN,-(SP)        ;ASSUME RUNNING STATE
3146 006430      TSTB     (R5)                 ;TEST FOR RUN OR HALT
3147 006432      BPL      10$                 ;BR IF RUNNING
3148 006434      MOV      #HLTED,(SP)         ;CHANGE TO HALTED
3149 006440      10$:    TYPEMES              ;TYPE 'RUNNING' OR 'HALTED'
3150 006442      TYPEMES #SOMMIS              ;TYPE ',SOMM IS '
3151 006450      MOV      #ISCLR,-(SP)        ;ASSUME CLEAR
3152 006454      BIT      #SOMMB,(R4)         ;TEST STATE OF SOMM
3153 006460      BEQ      20$                 ;BR IF CLEAR
3154 006462      MOV      #ISSET,(SP)        ;CHANGE TO SET
3155 006466      20$:    TYPEMES              ;TYPE 'SET' OR 'CLEAR'
3156 006470      TYPEMES #STPEQU              ;TYPE ',STEP='
3157 006476      MOV      #STINST,-(SP)       ;ASSUME INSTRUCTION STEP
3158 006502      TST      (R3)                 ;TEST FOR SINGLE INST STEP
3159 006504      BMI      30$                 ;BR IF SING INST STEP
3160 006506      MOV      #STBUS,(SP)        ;ASSUME BUS CYCLE STEP
3161 006512      BIT      #SBC,(R4)          ;TEST FOR BUS CYCLE STEP
3162 006516      BNE      30$                 ;BR IF BUS CYCLE
3163 006520      MOV      #STSTA,(SP)        ;ASSUME STATE STEP
3164 006524      BIT      #STS,(R4)          ;TEST FOR STATE STEP
3165 006530      BNE      30$                 ;BR IF STATE STEP
3166 006532      MOV      #NRMALL,(SP)       ;CHANGE TO NORMAL
3167 006536      30$:    TYPEMES              ;TYPE 'NONE', 'STAT', 'BUS', OR 'INST'
3168 006540      TYPEMES #CLKEQU              ;TYPE ',CLK='
3169 006546      MOV      #CLKNOR,-(SP)       ;ASSUME NORMAL FREQUENCY
3170 006552      BIT      #FREQ0!FREQ1,(R4)  ;TEST FOR NORMAL FREQ
3171 006556      BEQ      40$                 ;BR IF NORMAL FREQ
3172 006560      MOV      #CLKSLO,(SP)       ;ASSUME SLOW
3173 006564      BIT      #FREQ0,(R4)        ;TEST FOR SLOW
3174 006570      BEQ      40$                 ;BR IF SLOW
3175 006572      MOV      #CLKFAS,(SP)       ;ASSUME FAST
3176 006576      40$:    TYPEMES              ;TYPE 'NORM','SLOW','FAST'
3177 006600      TYPEMES #RADEQU,,CR          ;TYPE <CRLF><TAB>RAD=
3178 006606      MOV      #ORADIX,-(SP)      ;ASSUME OCTAL
3179 006612      TSTB     DEFRAD              ;TEST FOR OCTAL
3180 006616      BNE      50$                 ;BR IF OCTAL
3181 006620      MOV      #OHEX,(SP)        ;CHANGE TO HEX
3182 006624      50$:    TYPEMES              ;TYPE 'HEX' OR 'OCT'
3183 006626      TYPEMES #ADDEQU              ;TYPE ',ADD='
3184 006634      MOVB     DEFADS,R0           ;GET DEFAULT ADDRESS SPACE TO R0
3185 006640      ASL      R0                  ;TIMES 2
3186 006642      TYPEMES RAD!ST(R0)          ;TYPE 'GEN','CONS','INT','VIRT','FHYG','VBUS','IDBU'
3187 006650      TYPEMES #DATEQU              ;TYPE ',DAT='
3188 006656      MOVB     DEFLNH,R0          ;R0 GETS CODE FOR DATA LENGTH
3189 006662      ASL      R0
3190 006664      TYPEMES DAT!ST(R0)          ;TYPE 'BYTE','WORD','LONG','QUAD'
3191 006672      TYPEMES #FILLEQ              ;TYPE ',FILL='

```

ZZ-ESKAA-10.1 SHOW CONSOLE STATE
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 34-1
 SHOW CONSOLE STATE

```

3192 006700 012746 035547'      MOV    #TERFIL,-(SP)    ;CONVERT 'TERFIL' TO ASCII STRING
3193 006704 012746 000001      MOV    #1,-(SP)      ;TERFIL IS 1 BYTES LONG
3194 006710 004767 001522      JSR     PC,CVNTYP    ;GO CONVERT AND TYPE THE STRING
3195 006714                      TYPEMES #RELEQU      ;TYPE ',REL='
3196 006722 012746 022522'      MOV    #RELOCA,-(SP)   ;CONVERT CONTENTS OF RELOCATION REGISTER
3197 006726 012746 000004      MOV    #4,-(SP)      ;RELOC REG IS 4 BYTES LONG
3198 006732 004767 001500      JSR     PC,CVNTYP    ;CONVERT AND TYPE IT
3199 006736 000207                      RTS     PL
3200
3201 006740 021413' 021420' 021425' RADLST: .WORD  SPHY,SVIR,SGEH,SINT,SIDB,SCON,SVBU
      006746 021431' 021435' 021442'
      006754 021447'
3202 006756 021513' 021520' 021525' DATLST: .WORD  DBYT,DWRD,DLNG,QAD
      006764 021532'
3203
3204                      .DSABL  LSB

```

```

3206 .SBTTL SHOW VERSION INFO
3207
3208 006766 DOSHVR: ;DISPLAY VERSION OF PCS,WCS,FPLA, AND CONSOLE SOFTWARE
3209 ;INPUTS:
3210 ; 'PCSVER' = PCS VERSION #
3211 ; 'WPMVER' = WCS PRIMARY VERSION #
3212 ; 'WSCVER' = WCS SECONDARY VERSION #
3213 ; 'FLPVER' = FPLA VERSION #
3214 ;OUTPUTS: NONE
3215 ;EFFECTS: VERSION INFO DISPLAYED ON TERMINAL
3216
3217 006766 TYPEMES #PCSEQU,,CR ;TYPE'PCS='
3218 006774 012700 037752 MOV #PCSVER,R0 ;R0 GETS PCS VERSION PNTR
3219 007000 004767 000106 JSR PC,OUTASC ;OUTPUT PCS VERSION
3220 007004 TYPEMES #WCSEQU ;TYPE 'WCS='
3221 007012 012700 037753 MOV #WPMVER,R0 ;R0 GETS WCS PRIM VER PNTR
3222 007016 004767 000070 JSR PC,OUTASC ;OUTPUT WCS PRIM VER
3223 007022 TYPEMES #DASH ;TYPE '-'
3224 007030 012700 037754 MOV #WSCVER,R0 ;R0 GETS PNTR TO WCS SECONDARY VER
3225 007034 004767 000052 JSR PC,OUTASC ;OUTPUT WCS SEC VER
3226 007040 TYPEMES #FPLEQU ;TYPE 'FPLA='
3227 007046 012700 037755 MOV #FPLVER,R0 ;R0 GETS PNTR TO FPLA VER
3228 007052 004767 000034 JSR PC,OUTASC ;OUTPUT FPLA VERSION
3229 007056 TYPEMES #CONEQU ;TYPE 'spmesCON='
3230 007064 TYPEMES #CONVER ;TYPE CONSOLE VER #
3231 007072 122767 000001 037744' CMPB #GHOPT,MICOPT ;G & H FLOATING PRESENT?
3232 007100 001005 BNE 10$ ;BRANCH IF NO
3233 007102 TYPEMES #GHMES ;TYPE 'G&H PRESENT'
3234 007110 10$:
3235 ;
3236 ;TYPEMES #SPEC1,,CR ;SHOW THAT THIS IS A PRE-RELEASED VERSION
3237 ;G&H PRESENT TYPEOUT HAS BEEN COMMENTED OUT TO
3238 ;MAKE ROOM FOR THIS COMMENT. RESTORE WHEN WE
3239 ;RELEASE THIS CONSOLE.
3240 007110 000207 RTS PC
3241
3242 007112 OUTASC: ;SUBROUTINE TO CONVERT A BYTE TO HEX ASCII AND PRINT IT
3243 ;INPUTS: R0 IS POINTER TO BYTE TO CONVERT
3244
3245 007112 010046 MOV R0,-(SP) ;STACK POINTER TO BYTE
3246 007114 012746 000001 MOV #1,-(SP) ;STACK LENGTH IN BYTES
3247 007120 012746 000020 MOV #16,-(SP) ;STACK RADIX
3248 007121 004767 140022' JSR PC,CONVRT ;CONVERT TO ASCII STRING
3249 007130 TYPEMES ;TYPE STRING WHOSE POINTER IS ON STACK
3250 007132 000207 RTS PC
3251
3252
3253 007134 DOREBO: ;REBOOT CONSOLE
3254 007134 000177 140100' JMP @REBCON ;REBOOT CONSOLE, BUT NOT STAR
3255

```

```

3257          .SBTTL  SET DEFAULTS
3258
3259          .ENABL  LSB
3260
3261 007140      DOSTDF. ;SET DEFAULTS
3262 007140      012701 035423*      MOV      #DEFRAD,R1      ;POINT R1 TO DEFAULTS
3263 007144      012700 035420*      MOV      #CURRAD,R0      ;POINT R0 TO CURRENTS
3265 007150      012702 000002      MOV      #LNGLNH,R2      ;R2 USED FOR COUNTER OR CONSTANT
3266 007154      105767 013330      TSTB     TMPRAD      ;TEST FOR SET 'STANDARD' DEFAULTS
3267 007160      001004      BNE         20$      ;BR IF NOT TO SET STANDARDS
3268 007162      105021      CLRB      (R1)+      ;SET DEFAULTS TO STANDARD SETTINGS
3269 007164      110221      MOVB     R2,(R1)+      ;SET LENGTH TO LONG WORD
3270 007166      105021      CLRB      (R1)+      ;SET ADDRESS SPACE TO PHYSICAL
3271 007170      000406      BR         40$
3272
3273 007172      122021      20$:      CMPB     (R0)+,(R1)+      ;COMPARE CURRENT SETTING AGAINST DEFAULT
3274 007174      001402      BEQ         30$      ;BR IF THEY ARE THE SAME
3275 007176      114041      MOVB     -(R0),-(R1)      ;SET DEFAULT TO CURRENT
3276 007200      000774      BR         20$      ;DO SAME BYTE AGAIN TO UPDATE R0 AND R1
3277
3278 007202      005302      30$:      DEC         R2      ;CHECKED ALL THREE YET
3279 007204      002372      BGE         20$      ;BR IF NOT
3280 007206      000207      40$:      RTS         PC
3281
3282          .DSABL  LSB

```

```

3284 .SBTTL LOAD MICRO-DIAGNOSTIC MONITOR OR MICRO-DEBUGGER
3285
3286 .ENABL LSB
3287
3288 007210 DOTEST: ;LOAD MICRO-DIAGNOSTIC MONITOR
3289 ;R2-->TSTRUN
3290 007210 004712 JSR PC,(R2) ;TEST FOR CPU RUNNING
3291 007212 103435 BCS 10$ ;BR IF RUNNING
3292 007214 004267 000034 JSR R2,COMLOD ;CALL COMMON LOADER
3293 007220 051253 .RAD50 \MIC\ ;FILE NAME OF LOAD FILE
3294 007222 051646 .RAD50 \MON\
3295 007224 075273 .RAD50 \SYS\
3296
3297 007226 DOWCS: ;LOAD MICRO-DEBUGGER
3298 007226 004267 000022 JSR R2,COMLOD ;CALL COMMON LOADER
3299 007232 110113 .RAD50 \WCS\ ;FILE NAME FOR THIS LOAD
3300 007234 051646 .RAD50 \MON\
3301 007236 075273 .RAD50 \SYS\
3302
3303 007240 DOOVER: ;LOAD AN OVERLAY
3304 007240 OPEN$ #FILENM ;TRY TO OPEN FILE
3305 007250 103017 BCC 30$ ;BR IF OPEN SUCCESSFUL
3306 007252 000207 RTS PC
3307
3308 007254 COMLOD: ;COMMON OVERLAY LOADER
3309 007254 012701 022546' MOV #FILENM,R1 ;POINT R1 TO FILENAME BLOCK
3310 007260 012221 MOV (R2)+,(R1)+
3311 007262 012221 MOV (R2)+,(R1)+
3312 007264 012221 MOV (R2)+,(R1)+
3313 007266 F$OPEN #FILENM ;OPEN THE FILE
3314 007274 103005 BCC 30$ ;BR IF OPEN SUCCESSFUL
3315 007276 012600 MOV (SP)+,R0 ;R0 GETS ERROR CODE
3316 007300 004767 004314 JSR PC,TPERRM ;TYPE ERROR MESSAGE
3317 007304 005726 TST (SP)+
3318 007306 000207 10$: RTS PC
3319
3320 007310 105267 027447 30$: INCB NOCNLS ;INDICATE CONSOL.SYS OVERLAID.
3321 007314 042767 004000 026300 BIC #WCSPRES,FLAG ; MARK THAT WCS MUST BE RELOADED
3322 007322 000167 140042' JMP LODMIC ;XFER CONTROL TO OVERLAY LOADER
3323
3324 .DSABL LSB
  
```

```

3326          .SBTTL  WAIT FOR DONE,SET/CLR MEMORY MAPPING ENABLE
3327
3328          .ENABL  LSB
3329
3330 007326    DOWAIT: ;ENABLE A 'WAIT FOR DONE'
3331          ;R3-->'FLAG'
3332 007326    052713 000100    BIS      #WFDONE,(R3)
3333 007332    000167 023520    JMP      CLRRPT          ;CLEAR 'RPTFLG' THEN RETURN
3334
3335 007336    STCLMP: ;ROUTINE TO SET OR CLEAR MEMORY MAPPING ENABLE IN 'TBUF0'(ID 12)
3336          ;INPUTS:      C BIT IN SAME STATE AS MEMORY MAPPING BIT IS TO BE
3337          ;OUTPUTS:     IF<C BIT CLEAR ON ENTRY> THEN < MEMORY MAPPING DISABLED>
3338          ;              IF<C BIT SET ON ENTRY> THEN < MEMORY MAPPING ENABLED>
3339 007336    010046          MOV     R0,-(SP)
3340 007340    016700 027250    MOV     TBF0SV,R0          ;GET SAVED CONTENTS OF TBUF0(LSB'S)
3341 007344    042700 000001    BIC     #1,R0          ;CLEAR BIT 0
3342 007350    005500          ADC     R0
3343 007352    010146    10$:  MOV     R1,-(SP)
3344 007354    010246          MOV     R2,-(SP)
3345 007356    016701 027234    MOV     TBF0SV+2,R1        ;GET SAVED CONTENTS OF TBUF0(MSB'S)
3346 007362    012702 000022    MOV     #TBUF0,R2          ;R2 GETS ID ADDRESS OF TBUF0
3347 007366    004767 001414    JSR     PC,WRID12          ;WRITE R0,R1 TO TBUF0
3348 007372    012602          MOV     (SP)+,R2
3349 007374    012601          MOV     (SP)+,R1
3350 007376    012600          MOV     (SP)+,R0
3351 007400    000207          RTS     PC
3352
3353 007402    RESTMM: ;RESTORE MEMORY MAPPING ENABLE IN ID ADDRESS 12(HEX)
3354          ;INPUTS:      STATE OF MME SAVED IN 'TBF0SV'
3355          ;OUTPUTS:     NONE
3356          ;EFFECTS:     SAVED CONTENTS OF 'TBUF0'(ID 12) ARE REWRITTEN TO
3357          ;              'TBUF0'
3358 007402    010046          MOV     R0,-(SP)
3359 007404    016700 027204    MOV     TBF0SV,R0          ;R0 GETS SAVED TBUF0 CONTENTS(LSB'S)
3360 007410    000760          BR      10$
3361
3362          .DSABL  LSB

```

```

3364 .SBTTL CLOCK TICK REPORTING
3365
3366 007412 TYPTIC: ;ROUTINE TO REPORT CURRENT STATE OF STAR CLOCK
3367 ;INPUTS: R3-->'FLAG'
3368 ;OUTPUTS: NONE
3369 ;EFFECTS: TYPE 'CPT0,1,2,3' ON CONSOLE PRINTER
3370 ; IF<CPT0> THEN <UPC ALSO REPORTED>
3371 ; IF<CPT3> THEN <ACCELERATOR PC PRINTED>
3372 ; PROGRAM I/O MODE CONDITIONALLY CLEARED
3373 007412 004077 140054' JSR R0,@RSAVEP ;SAVE R0-R5, R3<-- POINTER TO 'FLAG'
3374 007416 052713 040000 BIS #IGNORE,(R3) ;INHIBIT THE CLOCK OFF MESSAGE
3375 007422 004767 000766 JSR PC,EXTPIO ;CHECK FOR PROGRAM I/O EXIT
3376 007426 112767 000060 012212 MOVB #0,CPTN+5 ;MAKE MESSAGE 'CPT0' INITIALLY
3377 007434 012701 173036 MOV #VBUSR,R1 ;POINT R1 TO VBUS REGISTER
3378 007440 111102 MOVB (R1),R2 ;R2 GETS CLOCK STATE BITS
3379 007442 100404 5$: BMI 10$ ;BR IF CPT0
3380 007444 105267 012176 INCB CPTN+5 ;INCREMENT THE CLOCK TICK NUMBER IN MESSAGE
3381 007450 106302 ASLB R2
3382 007452 000773 BR 5$
3383
3384 007454 10$: TYPEMES #CPTN,,CR ;TYPE 'CPT0,1,2,OR 3'
3385 007462 105711 TSTB (R1) ;CPT0?
3386 007464 100024 BPL 30$ ;BR IF NO
3387 007466 TYPEMES #UPCEQU ;TYPE ',UPC='
3388 007474 005067 027104 CLR EFFADR ;GET UPC FROM VBUS CHANNEL 0
3389 007500 004767 176352 JSR PC,EXUPC ;READ VBUS CHANNEL 0
3390 007504 042767 160000 027030 BIC #160000,DATAFR ;UPC IS IN LOWER 13 OF DATAFR
3391 007512 012746 036542' 20$: MOV #DATAFR,-(SP) ;CONVERT 'DATAFR' CONTENTS TO AN ASCII STRING
3392 007516 012746 000002 MOV #2,-(SP) ;STACK LENGTH IN BYTES
3393 007522 012746 000020 MOV #16,-(SP) ;FORCE THE RADIX TO HEX
3394 007526 004767 140022' JSR PC,CONVRT
3395 007532 TYPEMES ;TYPE THE UPC STRING
3396 007534 000573 BR REPLAC
3397
3398 007536 032711 000020 30$: BIT #CPT3,(R1) ;CPT3?
3399 007542 001570 REPLAC ;BR IF NOT
3400 007544 012700 173030 MOV #IDCNTR,R0 ;R0 POINTS TO ID CONTROL REG
3401 007550 012710 000026 MOV #ID16,(R0) ;SET ID ADDRESS
3402 007554 052710 000200 BIS #IDMANT,(R0) ;SET THE ID MAINT BIT
3403 007560 005067 026756 CLR DATAFR
3404 007564 013767 173006 026750 MOV @IDDATL,DATAFR ;GET ACCELERATOR PC
3405 007572 042767 177000 026742 BIC #177000,DATAFR ;CLEAR ALL EXCEPT 9 BITS(PX-03-00)
3406 007600 042710 000200 BIC #IDMANT,(R0)
3407 007604 TYPEMES #APCEQU ;TYPE 'APC='
3408 007612 000737 BR 20$

```

ZZ-ESKAA-10.1 CHECK FOR CLOCK STOP, WAIT FOR MICRO-RESPONSE
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 40
 CHECK FOR CLOCK STOP, WAIT FOR MICRO-RESPONSE

```

3410 .SBTTL CHECK FOR CLOCK STOP, WAIT FOR MICRO-RESPONSE
3411
3412 .ENABL LSB
3413
3414 007614 TSTCLK: ;ROUTINE TO CHECK FOR CLOCK STOPPED
3415 ;INPUTS: NONE
3416 ;OUTPUTS: C BIT SET IF CLOCK STOP DETECTED AND SINGLE BUS CYCLE ENABLED
3417 ; (PX0101-ONLY SET BUS CYCLE IF 'STS' OR 'SBC' NOT ASSERTED)
3418 ; C BIT CLEAR IF NO CLOCK STOP DETECTED, CLOCK MODE UNCHANGED
3419 007614 010446 MOV R4, -(SP)
3420 007616 012704 173032 MOV #MCR, R4
3421 007622 032714 000040 BIT #CLKSTD, (R4)
3422 007626 001425 BEQ 10$ ;BR IF CLOCK RUNNING
3423 007630 032767 040000 025764 BIT #IGNORE, FLAG ;CLOCK IS STOPPED. SEE IF WE ALREADY REPORTED IT.
3424 007636 001024 BNE 20$ ;BR IF ALREADY REPORTED
3425 007640 TYPEMES #TAB, CR ;TYPE <CRLF><TAB>
3426 007646 TYPEMES #CLOCKS ;TYPE 'CPU CLOCK STOPPED'
3427 007654 004767 177532 JSR PC, TYPTIC ;TYPE THE CLOCK STATE
3428 007660 TYPEMES #CRMES
3429 007666 032714 000006 BIT #STS!SBC, (R4) ;CLOCK ALREADY IN A STEP MODE?
3430 007672 001007 BNE 30$ ;BR TO EXIT IF YES
3431 007674 004767 174456 JSR PC, DOSSTB ;SET CLOCK TO BUS CYCLE MODE
3432 007700 000404 BR 30$
3433
3434 007702 042767 040000 025712 10$: BIC #IGNORE, FLAG ;ALLOW FUTURE CLOCK STOPS TO BE REPORTED
3435 007710 005727 20$: TST (PC)+
3436 007712 000261 30$: SEC
3437 007714 012604 MOV (SP)+, R4
3438 007716 000207 RTS PC
3439
3440
3441 007720 CWAIT: ;ROUTINE TO WAIT FOR A 'CONSOLE ACKNOWLEDGE' FROM STAR CPU
3442 ;IF<CONSOLE ACK NOT SEEN WITHIN TIME-OUT PERIOD>
3443 ;THEN<REPORT TIMEOUT, SET C BIT, RETURN>
3444 007720 010446 MOV R4, -(SP)
3445 007722 005004 CLR R4
3446 007724 105067 025667 CLR B TIMOUT ;CLEAR TIMEOUT AND ERROR FLAG
3447 007730 105737 173034 40$: TSTB #MCS ;WAIT FOR 'CNSLAK', USING R4 AS TIMEOUT
3448 007734 100765 BMI 20$ ;BR IF 'CNSLAK' IS SET (STAR HAS STOPPED)
3449 007736 005204 INC R4 ;PLUS ONE TO TIMEOUT COUNTER
3450 007740 001373 BNE 40$ ;BRANCH UNTIL R4 REACHES ZERO
3451 ;TIMED-OUT WAITING FOR STAR CPU RESPONSE
3452 ;EITHER MICRO-CODE IS NOT RUNNING OR IS SCREWED-UP
3453 007742 TYPEMES #TIMEOUT, CR ;TELL OPERATOR OF TIMEOUT
3454 007750 105267 025643 INCB TIMOUT ;SET TIMEOUT FLAG
3455 007754 000756 BR 30$
3456
3457 .DSABL LSB

```



```

3459 .SBTTL TEST FOR A MICRO-ROUTINE ERROR
3460
3461 .ENABL LSB
3462
3463 007756 TSTERR: ;ROUTINE TO READ MICRO-CODE FUNCTION STATUS REGISTER
3464 ;INPUTS: NONE
3465 ;EFFECTS: ID REGISTER 'D.SV'(2E) CONTAINS A CODE
3466 ; WHICH IS STATUS OF LAST FUNCTION
3467 ;OUTPUTS: C BIT SET IF ERROR MESSAGE TYPED
3468 007756 004077 140054' JSR R0,@RSAVEP ;SAVE R0-R5
3469 007762 105767 026773 TSTB XLOFLG ;ARE WE DOING AN 'X' BINARY LOAD
3470 007766 001056 BNE REPLAC ;EXIT, RESTORING REGISTERS, IF SO
3471 007770 012702 000056 MOV #DSV,R2 ;R5 GETS ADDRESS OF 'D.SV'
3472 007774 012703 036610' MOV #GOTID,R3 ;R3 POINTS TO WHERE ID BUS DATA GOES
3473 010000 004767 001036 JSR PC,READID ;READ D.SV TO 'GOTID'
3474 010004 005000 CLR R0
3475 010006 004767 000774 JSR PC,WRID12 ;CLEAR 'D.SV'
3476 010012 116700 026572 MOVB GOTID,R0 ;R0 GETS CODE RETURNED BY MICRO-ROUTINE
3477 010016 001440 BEQ 40$ ;BR IF CODE IS ZERO(SUCCESS)
3478 010020 020027 000013 CMP R0,#LASERR ;COMPARE CODE WE GOT TO HIGHEST POSSIBLE
3479 010024 101410 BLOS 20$ ;BR IF WITHIN RANGE
3480 010026 TYPEMES #UNKERR,,CR ;CODE OUT OF RANGE, TYPE UNKNOWN ERROR HALT MESSAGE
3481 010034 004767 003612 JSR PC,TYPEPC ;TYPE THE CODE WE GOT
3482 010040 105267 025553 10$: INCB TIMEOUT ;SET ERROR FLAG
3483 010044 000426 BR 50$
3484
3485 010046 006300 20$: ASL R0 ;TYPE ERROR MESSAGE USING ERROR CODE AS AN INDEX
3486 010050 TYPEMES TABMES(R0),,CR
3487 010056 020027 000014 CMP R0,#HLTINS*2 ;BREAK OUT OF HERE IF CODE IS 'HALT INSTRUCTION EXECUTED'
3488 010062 001416 BEQ 40$ ;TAKE NO ERROR EXIT
3489 010064 020027 000004 CMP R0,#CONERR*2 ;TEST FOR A 'CODE 2' ERROR
3490 010070 001003 BNE 30$ ;BR IF NOT A 'CODE 2'
3491 010072 004767 000366 JSR PC,TYPEIDR ;TO BE PRINTED ON CONSOLE
3492 010076 000760 BR 10$
3493
3494 010100 020027 000002 30$: CMP R0,#MEMFAL*2 ;TEST FOR A MEMORY MANAGEMENT FAILURE
3495 010104 001355 BNE 10$ ;BR IF NOT MEM-MAN FAILURE
3496 010106 116700 026477 MOVB GOTID+1,R0 ;TYPE THE CODE ASSOCIATED WITH THE ERROR
3497 010112 004767 003534 JSR PC,TYPEPC
3498 010116 000750 BR 10$
3499
3500 010120 005727 40$: TST (PC)+ ;CLEAR C BIT
3501 010122 000261 50$: SEC ;SET C BIT
3502 010124 REPLAC: ;RESTORE R0-R5. CALLED BY 'BR REPLAC' OR 'JMP REPLAC'
3503 010124 012605 MOV (SP)+,R5
3504 010126 012604 MOV (SP)+,R4
3505 010130 012603 MOV (SP)+,R3
3506 010132 012602 MOV (SP)+,R2
3507 010134 012601 MOV (SP)+,R1
3508 010136 012600 MOV (SP)+,R0
3509 010140 000207 RTS PC
3510
3511 .DSABL LSB

```

ZZ-ESKAA-10.1 TEST FOR A STAR CPU HALT, REPORT A HALT
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 42
 TEST FOR A STAR CPU HALT, REPORT A HALT

```

3513 .SBTTL TEST FOR A STAR CPU HALT, REPORT A HALT
3514
3515 010142 TSTHAL: ;TEST FOR A STAR CPU HALT
3516 ; 'SAWHLT' BIT OF FLAG IS SET IF HALT ALREADY REPORTED
3517 ;OUTPUTS: C BIT SET IF A HALT WAS SEEN AND REPORTED
3518 010142 105737 173034 TSTB a#MCS ;TEST FOR CPU MICRO-MACHINE IN WAIT LOOP
3519 010146 100016 BPL 20$ ;BR IF NOT HALTED
3520 010150 032737 000040 173032 BIT #CLKSTD,a#MCR ;TEST FOR CLOCK STOPPED
3521 010156 001012 BNE 20$ ;BR IF CLOCK STOPPED
3522 010160 032767 000002 025434 BIT #SAWHLT,FLAG ;TEST FOR THIS HALT 'READY REPORTED
3523 010166 001006 BNE 20$ ;BR IF IT WAS REPORTED PREVIOUSLY
3524 010170 004767 000014 JSR PC,REPHLT ;REPORT THIS HALT
3525 10174 TYPEMES #CRMES ;TYPE A CARRIAGE RETURN AND LINE FEED
3526 10202 022707 10$: CMP (PC)+,PC ;SET C BIT
3527 010204 000241 20$: CLC
3528 010206 900207 RTS PC
3529
3530
3531 010210 REPHLT: ;REPORT A STAR CPU HALT
3532 010210 004077 140054' JSR R0,a#SAVEP ;SAVE R0-R5, R3<--POINTER TO 'FLAG'
3533 010214 106746 MFPS -(SP) ;**
3534 010216 106427 000340 MTPS #340 ;BLOCK OUT LSI INTERRUPTS
3535 010222 042713 000004 BIC #IDSAVD,(R3) ;CLEAR THE 'ID SAVED' FLAG
3536 010226 105737 173014 TSTB a#RXDONE ;TEST FOR A CHARACTER IN 'TOID'(RXDB)
3537 010232 100010 BPL 10$ ;BR IF NO CHAR IN 'TOID'
3538 010234 052713 000004 BIS #IDSAVD,(R3) ;REMEMBER WE ARE SAVING 'TOID'
3539 010240 013767 173020 026330 MOV a#TOIDLO,SAVIDL ;SAVE 'TOID' REG
3540 010246 013767 173022 026324 MOV a#TOIDHI,SAVIDH
3541 010254 052737 000400 173032 10$: BIS #STRIND,a#MCR ;DISABLE STAR INTERRUPTS
3542 010262 042737 000140 173034 BIC #RDYIE!DNEIE,a#MCS ;DISABLE LSI INTS FORM 'RXDNE' AND 'RDYIE'
3543 010270 012702 000022 MOV #TBUFO,R2 ;SAVE ID REG 'TBUFO'
3544 010274 012703 036614' MOV #TBFOSV,R3
3545 010300 004767 000536 JSR PC,READID ;READ ID REG TO 'TBFOSV'
3546 010304 052767 000002 025310 BIS #SAWHLT,FLAG ;PREVENT THIS HALT FROM BEING REPORTED AGAIN
3547 010312 004767 000076 JSR PC,EXTPIO ;CHECK FOR PROGRAM I/O EXIT
3548 010316 004767 177434 20$: JSR PC,TSTERR ;TEST FOR AN ERROR HALT
3549 010322 TYPEMES #HLTMS,,CR ;TYPE THE HALT MESSAGE
3550 010330 026727 025062 003562' CMP WHATTODO,#DOHALT ;HERE AS RESULT OF HALT COMMAND?
3551 010336 001411 BEQ 25$ ;BR IF YES(SKIP SETTING UP AUTO-RESTART)
3552 010340 105767 025624 TSTB BOOTFL ;BOOTING?
3553 010344 001006 BNE 25$ ;BR AND SKIP FLAGGING AUTORESTART IF YES
3554 010346 052767 000020 025024 BIS #INITLD,TCONTL ;SET FLAG TO CAUSE AUTO-RESTART
3555 010354 016767 026230 012150 MOV GOTID,SAVCO ;SAVE THE 'HALT REASON' CODE( TO PASS INTO R12)
3556 010362 106426 25$: MTPS (SP)+ ;** RESTOR OLD PS
3557 010364 105067 025022 CLRB DEEXBY ;FORCE AN EXAMINE
3558 010370 004767 175404 JSR PC,EXDEPC ;EXAMINE THE STAR FC
3559 010374 103653 BCS REPLAC ;BR IF ERROR ON PC READ
3560 010376 012746 036542' MOV #DATAFR,-(SP) ;CONVERT PC TO ASCII STRING
3561 010402 012746 000004 MOV #4,-(SP)
3562 010406 004767 000024 JSR PC,CVNTYP ;CONVERT AND TYPE THE PC
3563 010412 000644 BR REPLAC
3564
3565 010414 EXTPIO: ;IF <NOT IN DISABLE> THEN <EXIT PROGRAM I/O MODE>
3566 010414 032737 000001 173034 BIT #LOCKD,a#MCS ;IN DISABLE POSITION?
3567 010422 001004 BNE 10$ ;BR IF YES

```



```

3589 010464          TYPIDR: ;ROUTINE TO TYPE OUT A LIST OF ID BUS ADDRESSES
3590                  ;INPUTS:      RO POINTS TO A BYTE LIST OF ID BUS ADDRESSES.
3591                  ;              LIST IS TERMINATED BY A 0
3592 010464 004077 140054' JSR      RO,@RSAVEP      ;SAVE REGISTERS,POINT R3 TO 'FLAG'
3593 010470 052713 000040 BIS      #SAWERR,(R3)    ;REMEMBER A TYPE 2 ERROR OCCURRED
3594 010474 012700 010606' MOV      #IDTABL,RO    ;POINT RO TO LIST OF ADDRESSES TO READ
3595 010500 005001 10$: CLR      R1              ;USE R1 TO COUNT REGISTERS TYPED PER LINE
3596 010502          TYPMES  #TAB,,CR          ;TYPE A CRLF AND A TAB
3597 010510 111067 026066 20$: MOVB    (RO),IDTEMP    ;GET AN ID REGISTER ADDRESS
3598 010514 001603 BEQ      REPLAC            ;BR IF AT END OF LIST
3599 010516          TYPMES  #OPNPAR          ;TYPE "<<ADDRESS>> <CONTENTS> "
3600 010524 012746 036602' MOV      #IDTEMP,-(SP)    ;CONVERT ID REG ADDRESS AND TYPE IT
3601 010530 012746 000001 MOV      #1,-(SP)
3602 010534 004767 177676 JSR      PC,CVNTYP      ;CONVERT ADDRESS TO ASCII AND TYPE
3603 010540          TYPMES  #CLSPAR          ;TYPE CLOSING PAREN AND ONE SPACE
3604 010546 012703 036610' MOV      #GOTID,R3        ;R3 POINTS TO WHERE ID DATA GOES
3605 010552 112002 MOVB    (R0)+,R2          ;R2 GETS ADDRESS
3606 010554 004767 000262 JSR      PC,READID      ;GET CONTENTS OF ID REGISTER ADDRESSED BY R2
3607 010560 012746 036610' MOV      #GOTID,-(SP)    ;NOW CONVERT CONTENTS TO ASCII STRING
3608 010564 012746 000004 MOV      #4,-(SP)
3609 010570 004767 177642 JSR      PC,CVNTYP      ;CONVERT CONTENTS TO ASCII AND TYPE
3610 010574 005201 INC      R1              ;UPDATE ITEMS PER LINE COUNTER
3611 010576 020127 000003 CMP      R1,#3          ;CHECK FOR 3 ITEMS TYPED ON ONE LINE
3612 010602 002742 BLT      20$              ;BR IF LESS THAN 3 ON THIS LINE
3613 010604 000735 BR       10$              ;START A NEW LINE
3614
3615
3616 010606 014 022 023 IDTABL: .BYTE  CESREG,TBUF0,TBUF1,SBIERR,SBIADD,CACPAR,0
      010611 031 032 036
      010614 000
3617          .EVEN
3618
3619 010616          TSTTY2: ;TEST FOR A CODE 2 MICRO-ERROR HAVING OCCURRED. IF IT HAS,
3620                  ;CLEAR OUT SOME ID REGISTER ERROR BITS
3621                  ;INPUTS:      'SAWERR' BIT OF 'FLAG' SET IF TYPE2 ERROR OCCURRED
3622                  ;OUTPUTS:    ID REGS 13,19, AND 1E ARE CLEARED OF ERRORS IF 'SAWERR'
3623                  ;              BIT OF 'FLAG' WAS SET ON ENTRY
3624 010616 004077 140054' JSR      RO,@RSAVEP      ;SAVE R0-R5,R3 GETS POINTER TO 'FLAG'
3625 010622 032713 000040 BIT      #SAWERR,(R3)    ;TEST FOR 'CODE 2' ERROR HAVING OCCURRED
3626 010626 001437 BEQ      20$              ;BR IF NO CODE 2 ERROR OCCURRED
3627 010630 042713 000040 BIC      #SAWERR,(R3)    ;CLEAR THE BIT THAT REMEMBERS THE ERROR
3628 010634 012705 000023 MOV      #TBUF1,R5        ;FIRST CLEAR ID REGS 13 AND 1E BY WRITING CONTENTS TO THEMSE
3629 010640 012704 036610' 5$: MOV      #GOTID,R4        ;R4 GETS A USEFUL POINTER
3630 010644 010502 MOV      R5,R2
3631 010646 010403 MOV      R4,R3
3632 010650 004767 000166 JSR      PC,READID
3633 010654 014301 MOV      -(R3),R1      ;R1 GETS THE MSB'S OF ID REG READ
3634 010656 014300 MOV      -(R3),R0      ;R0 GETS LSB'S
3635 010660 004767 000072 JSR      PC,WRITID
3636 010664 020527 000023 CMP      R5,#TBUF1      ;TEST FOR FIRST OR SECOND PASS THRU
3637 010670 001003 BNE      10$              ;BR IF SECOND PASS
3638 010672 012705 000036 MOV      #CACPAR,R5      ;GET REG 1E NEXT
3639 010676 000760 BR       5$
3640
3641 010700          10$:          ;CLEAR ID REG 'SBIERR'(19) BY READING CONTENTS TO A TEMPORARY LOCATION,

```

Address	Hex	Hex	Hex	Hex	Instruction	Comment
3642						;SETTING BITS IN THAT TEMPORARY CORRESPONDING TO 'CP TIMEOUT','CRD',AND 'RDS'
3643						;THEN WRITE THE TEMP BACK TO 'SBIERR'
3644	010700	012702	000031		MOV	#SBIERR,R2
3645	010704	010403			MOV	R4,R3 ;POINT R3 TO 'GOTID'
3646	010706	004767	000130		JSR	PC,READID
3647						;NOW GOTID HOLDS CONTENTS READ FROM 'SBIERR'
3648						;SET PROPER BITS (14,13,12)
3649	010712	014361			MOV	-(R3),R1 ;R1 GETS MSB'S OF ID REG READ
3650	010714	014300			MOV	-(R3),R0 ;R0 GETS LSB'S
3651	010716	052700	070000		BIS	#70000,R0
3652	010722	004767	000030		JSR	PC,WRITID
3653	010726	000167	177172	20\$:	JMP	REPLAC

```

3655 .SBTTL PUSH MICRO-STACK,READ/WRITE ID BUS REGISTERS
3656
3657 .ENABL LSB
3658
3659 010732 PUSHU: ;ROUTINE TO PUSH A WORD ON THE MICRO-STACK
3660 ;INPUTS: R0 IS LSB'S OF WORD TO PUSH
3661 ;OUTPUTS: C BIT SET IF CLOCK IS STOPPED
3662 010732 010246 MOV R2,-(SP)
3663 010734 042700 160000 BIC #160000,R0 ;CLEAR UNUSED MICRO-ADDRESS BITS
3664 010740 012702 000040 MOV #IDAUST,R2 ;PUT ID BUS ADDRESS OF MICRO-STACK IN R2
3665 010744 005001 CLR R1
3666 010746 004757 000004 JSR PC,WRITID
3667 010752 0126J2 MOV (SP)+,R2
3668 010754 000207 RTS PC
3669
3670 010756 WRITID: ;ROUTINE TO WRITE TO AN ID BUS ADDRESS
3671 ;INPUTS: R0,R1 ARE DATA TO WRITE(R0 IS LSB)
3672 ; R2 IS ID BUS ADDRESS
3673 ;EFFECT: 'TOID LO' GETS R0
3674 ; 'TOID HI' GETS R1
3675 ; 'ID ADDRESS' GETS R2
3676 ; R2=R2 'AND' 177700
3677 ; ID REG SPECIFIED BY 'ID ADDRESS' GETS 'TOID'
3678 ;OUTPUTS: IF ID ADDRESS 12 IS REFERENCED(.,JF0) THEN THE
3679 ; DATA WRITTEN IS SAVED IN 'TBUF0'
3680 010756 042702 177700 BIC #177700,R2 ;CLEAR ALL EXCEPT ID ADDRESS IN R2
3681 010762 004767 000102 JSR PC,TSTCST ;TEST FOR CLOCK STOP
3682 010766 103424 BCS 50$ ;BR IF CLOCK STOP OCCURRED
3683 010770 020227 000022 CMP R2,#TBUF0 ;CHECK FOR A REFERENCE TO 'TBUF0'
3684 010774 001004 BNE WRID12
3685 010776 010067 025612 MOV R0,TBUF0SV ;CHANGE OUR SAVED COPY OF 'TBUF0'
3686 011002 010167 025610 MOV R1,TBUF0SV+2 ;FOR USE WHEN DOING PHYSICAL OR VIRTUAL MEMORY REFERENCES
3687 011006 052702 100100 WRID12: BIS #IDCYCL,IDWRIT,R2 ;SET 'ID CYCLE' AND 'ID WRITE'
3688 011012 010037 173020 MOV R0,a#TOIDLO
3689 011016 010137 173022 MOV R1,a#TOIDHI
3690 011022 010237 173030 MOV R2,a#IDCNTL
3691 011026 042737 000100 173030 BIC #IDWRIT,a#IDCNTL
3692 011034 RTCCLR:
3693 011034 005727 30$: TST (PC)+ ;CLEAR C BIT
3694 011036 000261 40$: SEC ;SET C BIT
3695 011040 000207 50$: RTS PC
3696
3697 011042 READID: ;ROUTINE TO READ AN ID BUS REGISTER.(CLOCK RUNNING)
3698 ;INPUTS: R2 IS ID BUS ADDRESS
3699 ; R3 POINTS TO WHERE ID DATA GOES
3700 ;OUTPUTS: C BIT CLEAR
3701 ; (R3)=LOWER 16 BITS OF ID REGISTER
3702 ; 2(R3)=UPPER 16 BITS OF ID REGISTER
3703 ; R3 ON EXIT = R3 ON ENTRY PLUS 4
3704 ; R2 IS Clobbered
3705
3706 011042 042702 177700 BIC #177700,R2 ;CLEAR OUT ALL EXCEPT ADDRESS IN R2
3707 011046 052702 100000 BIS #IDCYCL,R2 ;SET ID CYCLE BIT IN R2
3708 011052 010237 173030 MOV R2,a#IDCNTL ;PUT ADDRESS IN ID CONTROL REG AND START READ
3709 011056 013723 173024 MOV a#FMIDLO,(R3)+ ;DATA NOW IN 'FMID'

```

ZZ-ESKAA-10.1 PUSH MICRO-STACK,READ/WRITE ID BUS REGISTERS
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 44-1
 PUSH MICRO-STACK,READ/WRITE ID BUS REGISTERS

```

3710 011062 013723 173026      MOV    a#FMIDHI,(R3)+
3711 011066 000762              BR      30$
3712
3713 011070      TSTCST: ;TEST FOR A CLOCK STOP
3714                  ;IF<CPU CLOCK IS STOPPED>
3715                  ;      THEN< 1)TYPE <CRLF><TAB>:CPU CLOCK STOPPED,COMMAND ABORTED
3716                  ;      2) SET C BIT,RETURN >
3717                  ;      ELSE< CLEAR C BIT,RETURN>
3718 011070 032737 000040 173032  BIT     #CLKSTD,a#MCR ;TEST CLOCK STOPPED BIT
3719 011076 001756              BEQ     30$ ;BR IF CLOCK RUNNING
3720 011100              TYPEMES #CLKERR,,CR
3721 011106              TYPEMES #CLOCKS
3722 011114      TYPCAD: TYPEMES #CANTDO
3723 011122 000745              BR      40$
3724
3725                  .DSABL  LSB

```

Z7-ESKAA-10.1 TEST FOR STAR CPU RUNNING
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 45
 TEST FOR STAR CPU RUNNING

```

3727          .SBTTL  TEST FOR STAR CPU RUNNING
3728
3729
3730 011124          TSTRUN: ;IF<HALTED> THEN <RETURN C BIT CLEAR> ELSE <RETURN C BIT SET>
3731 011124 105737 173034    TSTB  a#MCS          ;TEST FOR 'CPU IN WAIT LOOP'
3732 011130 100741          BMI   RTCCLR          ;BRANCH IF CPU IS HALTED
3733          ;CPU IS NOT HALTED(I.E. IN WAIT LOOP), CHECK FOR
3734          ;RUN BIT CLEAR(MEANS CPU MICRO-CODE IS NOT RUNNING)
3735 011132 032737 000400 173034    BIT   #RUNBIT,a#MCS
3736 011140 001735          BEQ   RTCCLR          ;BR IF TIMED-OUT(ALLOW FUNCTION)
3737 011142          TYPEMES #STRRUN,,CR        ;TELL OPERATOR CPU IS NOT IN WAIT LOOP
3738 011150 000761          BR    TYPCAD         ;TYPE 'COMMAND ABORTED', THEN RETURN
3739
3740          ;MICRO-CODE ERROR MESSAGE INDEX TABLE
3741          TABMES=-BASE-2
3742 011152 020737'          .WORD  MEMMAN
3743 011154 020765'          .WORD  CONSER
3744 011156 021013'          .WORD  RESCOM
3745 011160 021032'          .WORD  INSTIV
3746 011162 021052'          .WORD  CPDBLE
3747 011164 021145'          .WORD  HLINST
3748 011166 021075'          .WORD  ILIEVC
3749 011170 021113'          .WORD  NOWCSU
3750 011172 021130'          .WORD  EINTPE
3751 011174 021171'          .WORD  ERRCHM
3752 011176 021203'          .WORD  ERRPRG

```



```

3754 .SBTTL TEST FOR A MICRO-MACHINE TIME OUT
3755
3756 011200 TSTTMO: ;TEST FOR A STAR CPU MICRO-MACHINE TIME OUT
3757 ;
3758 ;INPUTS: NONE
3759 ;
3760 ;OUTPUTS: C BIT SET IF MICRO-MACHINE TIME OUT
3761 ;
3762 ;PROCESS: THERE IS AN 'INTERRUPT STROBE TIME OUT' CIRCUIT
3763 ; ON THE CONSOLE INTERFACE BOARD, THAT MEASURES THE
3764 ; TIME BETWEEN SUCCESSIVE STAR 'INTERRUPT STROBES'.
3765 ; A FAILURE TO STROBE INTERRUPTS WITHIN A CERTAIN
3766 ; MAXIMUM TIME PERIOD INDICATES A PROBLEM IN THE
3767 ; MICRO-MACHINE, AND WILL CAUSE A BIT TO CLEAR IN
3768 ; THE CONSOLE INTERFACE'S 'MCS' REGISTER. IF THIS
3769 ; ROUTINE DETECTS SUCH A TIME OUT, THE FOLLOWING
3770 ; ACTION IS TAKEN:
3771 ;
3772 ; 1) THE CONSOLE DISPLAYS: 'MICRO-MACHINE TIME OUT'
3773 ; 2) THE CONSOLE SETS A FLAG TO PREVENT REPEATING
3774 ; THIS MESSAGE UNTIL A NON-TIME OUT CONDITION IS SEEN.
3775 ; 3) IF AUTO-RESTART IS ENABLED, THE CONSOLE WILL
3776 ; MOMENTARILY STOP THE STAR CPU CLOCK AND 'SNAP SHOT'
3777 ; THE STAR MICRO-PC. UPON RETURN FROM THIS ROUTINE
3778 ; A STAR AUTO-RESTART SEQUENCE WILL BE PERFORMED.
3779 ;
3780 ;TIME OUT CONDITION DEFINITION:
3781 ; IF<<(STAR NOT HALTED)'AND'<(CLOCK RUNNING)'AND<(INT TIMEOUT BIT=0)>>
3782 ; THEN<TIME OUT IS INDICATED>
3783 ;
3784 011200 000241 CLC
3785 011202 032737 000400 173034 BIT #RUNBIT,a#MCS ;INTERRUPT TIME-OUT BIT CLEAR?
3786 011210 001044 BNE 20$ ;BR IF NOT TO EXIT
3787 011212 105737 173034 TSTB a#MCS ;STAR HALTED?
3788 011216 100437 BMI 10$ ;BR IF YES AND EXIT(CLEARING TIMEOUT SEEN FLAG)
3789 011220 032737 000040 173032 BIT #CLKSTD,a#MCR ;STAR CLOCK STOPPED?
3790 011226 001033 BNE 10$ ;BR IF YES AND EXIT
3791 011230 105767 011257 TSTB SAWTMO ;HAS THIS TIMEOUT BEEN REPORTED?
3792 011234 001032 BNE 20$ ;BR IF YES AND EXIT
3793 011236 105267 011251 INCB SAWTMO ;REMEMBER WE SAW A TIMEOUT
3794 011242 TYPEMES #MMTMOU,,CR ;TYPE THE TIME OUT MESSAGE
3795 011250 105767 037751' TSTB AUTFLG ;AUTORESTART ENABLED?
3796 011254 001416 BEQ 5$ ;BR IF NOT AND EXIT
3797 011256 052767 000020 024114 BIS #INITLD,TCONTL ;SET SOFT AUTORESTART FLAG TO CAUSE AR LATER
3798 011264 052737 000002 173032 BIS #SBC,a#MCR ;STOP THE CPU CLOCK
3799 011272 004767 176114 JSR PC,TYPTIC ;SNAP SHOT THE MICRO-PC
3800 011276 042737 000002 173032 BIC #SBC,a#MCR ;CLEAR SINGLE BUS CYCLE BIT
3801 011304 052737 000001 173032 BIS #PROCED,a#MCR ;RESTART CLOCK
3802 011312 000261 5$: SEC ;SET C BIT TO INDICATE TIME OUT SEEN
3803 011314 000402 BR 20$ ;EXIT
3804
3805 011316 105067 011171 10$: CLRB SAWTMO ;INITIALIZE 'SAW A TIMEOUT' FLAG
3806 011322 000207 20$: RTS PC

```

```

3808 .SBTTL PCS,WCS,FPLA VERSION CHECKING
3809
3810 011324 TSTVER: ;MAKE VERSION COMPATIBILITY CHECK OF MICRO-SOFTWARE
3811 ;TEST FOR:
3812 ; 0) IS WCS LOADED?
3813 ; 1) WCS PRIMARY VERSION = FPLA VERSION
3814 ; 2) UPPER 2 BITS OF WCS SECONDARY VERSION = PCS VERSION
3815 ;
3816 ; IF<0 FALSE> THEN <TYPE WARNING, EXIT C BIT SET>
3817 ; IF<1 AND 2 TRUE> THEN <EXIT, C BIT CLEAR>
3818 ; IF<1 FALSE & 2 TRUE> THEN <TYPE WARNING,EXIT C BIT CLEAR>
3819 ; IF<2 FALSE> THEN <TYPE WARNING, EXIT C BIT SET>
3820
3821 011324 010046 MOV R0,-(SP)
3822 011326 032767 004000 024266 BIT #WCSPRES,FLAG ;IS WCS LOADED?
3823 011334 001424 BEQ 15$ ; BRANCH IF NC
3824 011336 126767 037753' 037755' CMPB WPMVER,FPLVER ;WCS PRIM VER = FPLA VER?(TEST 1)
3825 011344 001403 BEQ 10$ ;BR IF WCS MATCHES FPLA
3826 011346 TYPEMES #WCNEFP,,CR ;TYPE 'WARNING-WCS & FPLA VER MISMATCH'
3827 011354 116700 037754' 10$: MOVW WSCVER,R0 ;R0 GETS WCS SEC VER
3828 011360 042700 177717 BIC #177717,R0 ;CLR ALL EXCEPT PCS VER
3829 011364 000241 CLC ;GET PCS VERSION BITS TO LSB'S OF R0
3830 011366 106100 ROLB R0
3831 011370 106100 ROLB R0
3832 011372 106100 ROLB R0
3833 011374 106100 ROLB R0
3834 011376 106100 ROLB R0
3835 011400 120067 037752' CMPB R0,PCSVR ;WCS SEC VER = PCS VER?(TEST 2)
3836 011404 001404 BEQ 20$ ;BR IF MATCH
3837 011406 15$: TYPEMES #WCNEPC,,CR ;TYPE 'FATAL-WCS & PCS VER MISMATCH'
3838 011414 022707 CMP (PC)+,PC ;SET C BIT, SKIP NEXT INSTRUCTION
3839 011416 000241 CLC
3840 011420 012600 MOV (SP)+,R0
3841 011422 000207 RTS PC
3842
3843
3844 011424 GETVER: ;COLLECT VERSIONS OF WCS,PCS, AND FPLA
3845 ;PCS VERSION IN LOWER 2 BITS OF ID ADDRESS FIELD FROM LOC 111 IN MICRO-STORE
3846 ;WCS PRIMARY VERSION IN ID ADDRESS OF LOC 1111
3847 ;WCS SECONDARY VERSION IN ID ADDRESS OF LOC 1112
3848 ;FPLA VERSION IS LOWER 6 BITS OF NEXT MICRO-PC AFTER F80
3849 ;MICRO CODE OPTION FLAG (KE780) IS BIT 0 OF NEXT MICRO PC AFTER 085
3850 ;STATE OF STAR: CLOCK RUNNING, MICRO-MACHINE IN CONSOLE SERVICE LOOP
3851 ; BOTH BEFORE AND AFTER VERSION COLLECTION
3852 ;
3853 ; GET PCS AND WCS VERSIONS
3854 ;
3855 011424 012700 000421 MOV #PCVRS,R0 ;R0 GETS ADDRESS OF PCS VERSION INSTRUCTION
3856 011430 004767 000256 JSR PC,RDIDAD ;R5 GETS ID ADDRESS FIELD
3857 011434 042705 177774 BIC #177774,R5 ;CLEAR UNUSED BITS
3858 011440 110567 037752' MOVW R5,PCSVR ;SAVE PCS VERSION
3859 011444 012700 010421 MOV #WCVRS,R0 ;R0 GETS PNTR TO WCS PRIM VER INSTRUCTION
3860 011450 004767 000236 JSR PC,RDIDAD ;R5 GETS ID ADDRESS FIELD
3861 011454 110567 037753' MOVW R5,WPMVER ;SAVE WCS PRIMARY VER
3862 011460 012700 010421 MOV #WCVRS,R0

```

```

3863 011464 005200          INC      R0          ;COMPUTE ADDRESS OF WCS SEC VER INSTRUCTION
3864 011466 004767 000220    JSR      PC,RDIDAD ;R5 GETS ID ADDRESS FIELD
3865 011472 110567 937754'   MOV     R5,WSCVER ;SAVE WCS SECONDARY VER
3866
3867          ; GET FPLA VERSION AND OPTION FLAG
3868
3869 011476 012700 000377     MOV     #377,R0      ;R0 GETS POINTER TO CONSOLE ROUTINE
3870 011502 004767 177224     JSR      PC,PUSHU      ;PUT ON MICRO STACK
3871 011506 012700 000205     MOV     #MOPTFL,R0    ;R0 GETS PTR TO KE780 PRESENT FLAG
3872 011512 004767 177214     JSR      PC,PUSHU      ;PUT ON MICRO STACK
3873 011516 012700 007600     MOV     #FPVERS,R0    ;R0 GETS PTR TO FPLA VER INSTRUCTION
3874 011522 004767 177204     JSR      PC,PUSHU      ;PUSH R0 ON STAR'S MICRO-STACK
3875
3876          ; ALL 3 ADDRESSES ON MICRO STACK, FIRST GET FPLA VERSION
3877
3878 011526 012704 173032     MOV     #MCR,R4
3879 011532 004767 172620     JSR      PC,DOSSTB      ;SET SINGLE BUS CYCLE
3880 011536 052714 002200     BIS     #MAINTR!ROMNOP,(R4)
3881
3882 011542 052714 000001     BIS     #PROCED,(R4)    ;SET MAINTENANCE RETURN BIT & ROM NOP
3883 011546 052714 000001     BIS     #PROCED,(R4)    ;CAUSE 1 CYCLE(TO ENABLE MAINT RET)
3884
3885 011552 052714 000001     BIS     #PROCED,(R4)    ;ALLOW ECO TO OCCUR LATCHING NEW ADDRESS
3886 011556 105067 023630     CLRP    DEEXBY      ;IN ECO ADDRESS LATCH
3887 011562 005067 025016     CLP     EFFADR      ;LATCH NEW ADDRESS IN UPC LATCH
3888 011566 010446           MOV     R4,-(SP)      ;FORCE EXAMINE
3889 011570 004767 174262     JSR      PC,EXUPC      ;PREPARE TO READ MICRO-PC
3890 011574 012604           MOV     (SP)+,R4      ;SAVE R4
3891 011576 116767 024740 037755' MOV     DATAFR,FPLVER ;CALL MICRO-PC EXAMINE RTN
3892 011604 142767 000300 037755' BICB    #300,FPLVER    ;RESTORE R4
3893
3894          ; NOW GET THE G & H OPTION FLAG FROM THE FPLA
3895
3896 011612 052714 002000     BIS     #MAINTR,(R4)    ;GET FPLA VERSION
3897 011616 052714 000001     BIS     #PROCED,(R4)    ;SCRATCH UNUSED BITS
3898 011622 052714 000001     BIS     #PROCED,(R4)
3899
3900 011626 052714 000001     BIS     #PROCED,(R4)    ;SET MAINTENANCE RETURN BIT
3901 011632 010446           MOV     R4,-(SP)      ;CAUSE 1 CYCLE(TO ENABLE MAINT RET)
3902 011634 004767 174216     JSR      PC,EXUPC      ;ALLOW ECO TO OCCUR LATCHING NEW ADDRESS
3903 011640 012604           MOV     (SP)+,R4      ;IN ECO ADDRESS LATCH
3904 011642 116767 024674 037744' MOV     DATAFR,MICOPT ;LATCH NEW ADDRESS IN UPC LATCH
3905 011650 142767 177776 037744' BICB    #^C<OPTMSK>,MICOPT ; ...
3906
3907          ; NOW RETURN TO THE CONSOLE WAIT LOOP
3908
3909 011656 052714 002000     BIS     #MAINTR,(R4)    ;SET MAINTENANCE RETURN BIT
3910 011662 042714 000202     BIC     #SBC!ROMNOP,(R4) ;CLEAR SINGLE BUS CYCLE & ROM NOP
3911 011666 052714 000001     BIS     #PROCED,(R4)    ;FREE RUN CLOCK
3912 011672 052767 000040 023722 BIS     #SAWERR,FLAG    ;THIS WILL FORCE ERROR CLEARING IN 'TSTTY2'
3913 011700 000167 176712     JMP      TSTTY2      ;CLEAR SBI ERROR REGISTERS(ID BUS SPACE)
3914
3915 011704          00C          004          010  SIZTBL: .BYTE 0,4,10,14,20 ;0,1K,2K,3K,4K IF IN UPPER BYTE OF A WORD
3916 011707          014          020
          .EVEN

```

ZZ-ESKAA-10.1 PCS,WCS,FPLA VERSION CHECKING
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 47-2
PCS,WCS,FPLA VERSION CHECKING

G 7

20-MAY-1986

Fiche 1 Frame G7

Sequence 84

3917

```
3919 .SBTTL READ ID BUS REGISTER ROUTINE
3920 011712 RDIDAD: ;SET MICRO-PC AND READ ID ADDRESS FIELD
3921 ;INPUTS: R0 IS MICRO-ADDRESS TO USE
3922 ;OUTPUTS: R5 IS ID ADDRSS FROM MICRO-INST
3923
3924 011712 004767 177014 JSR PC,PUSHU ;PUSH R0 ONTO MICRO-STACK
3925 011716 012704 173032 MOV #MCR,R4
3926 011722 004767 172430 JSR PC,DOSSTB ;ENABLE SINGLE BUS CYCLE MODE
3927 011726 052714 000200 BIS #ROMNOP,(R4) ;ASSERT ROM-NOP
3928 011732 052714 002000 BIS #MAINTR,(R4) ;ASSERT MAINTENANCE RETURN
3929 011736 052714 000001 BIS #PROCED,(R4) ;1 CYCLE TO ENABLE MAINT RET
3930 011742 042714 000200 BIC #ROMNOP,(R4) ;DISABLE ROM NOP
3931 011746 052714 000001 BIS #PROCED,(R4) ;1 CYCLE TO PERFORM MAINT RET
3932 011752 042714 000002 BIC #SBC,(R4) ;CLEAR SINGLE BUS CYCLE
3933 011756 113705 173031 MOVB @#IDCNTL+1,R5 ;R5 GETS ID ADDRESS BITS
3934 011762 005105 COM R5 ;COMP BITS BECAUSE HW REVERSES SENSE
3935 011764 042705 177700 BIC #177700,R5 ;CLEAR UNUSED BITS
3936 011770 052714 000001 BIS #PROCED,(R4) ;FREE RUN CLOCK
3937 011774 000207 RTS PC
```

```

3939 .SBTTL FILENAME CONVERSION TO RAD50
3940
3941 .ENABL LSB
3942
3943 011776 XLATFN: ;TRANSLATE A FILENAME FROM ASCII TO RAD50
3944 ;INPUTS: R4-->ASCII FILENAME STRING
3945 ;OUTPUTS: IF C BIT SET:
3946 ; 'FILENM' CONTAINS FILENAME IN RAD50
3947 ; R4-->FIRST CHARACTER BEYOND FILENAME IN INPUT STRING
3948 ;
3949 ; IF C BIT CLEAR:
3950 ; FILE NAME COULD NOT BE TRANSLATED
3951 ; R4 UNCHANGED
3952
3953 011776 010446 MOV R4,-(SP)
3954 012000 042767 002000 023614 BIC #USEDEF,FLAG ;CLEAR USE DEFAULT FLAG
3955 012006 012700 022546' MOV #FILENM,R0 ;POINT R0 TO FILENAME BLOCK
3956 012012 005020 CLR (R0)+
3957 012014 005020 CLR (R0)+
3958 012016 005020 CLR (R0)+ ;FILE NAME IS INITIALLY BLANKS
3959 012020 005001 CLR R1 ;R1 IS A COUNTER
3960 012022 105067 010457 CLRB CNVTDN ;CLEAR 'CONVERSION DONE' FLAG
3961 012026 105767 010453 10$: TSTB CNVTDN ;TEST CONVERT DONE FLAG
3962 012032 001121 BNE 50$ ;BR IF CONVERSION DONE
3963 012034 004767 002746 JSR PC,TESTND ;CHECK FOR A DELIMITER IN THE INPUT STRING
3964 012040 103410 BCS 1$ ;BR IF NO DELIMITER
3965 012042 105267 010437 INCB CNVTDN ;REMEMBER CONVERSION IS DONE
3966 012046 005701 TST R1 ;NO FILE NAME?
3967 012050 001061 BNE 30$ ;BRANCH IF FILE NAME SPECIFIED
3968 012052 052767 002000 023542 BIS #USEDEF,FLAG ;SET USE DEFAULT FLAG
3969 012060 000455 BR 30$ ;GO LEFT JUSTIFY AND ZERO FILL
3970
3971 012062 121427 000056 1$: CMPB (R4),#56 ;CHECK FOR A PERIOD
3972 012066 001451 BEQ 9$ ;BR IF PERIOD. LEFT JUSTIFY FILE NAME
3973 012070 005201 INC R1 ;ADD 1 TO POSITION COUNTER
3974 012072 010146 MOV R1,-(SP) ;SAVE R1 TEMPORARILY
3975 012074 005002 CLR R2 ;SET CONVERSION POINTER TO PROPER WORD OF FILENAME BLOCK
3976 012076 162701 000003 2$: SUB #3,R1
3977 012102 003402 BLE 3$
3978 012104 005202 INC R2
3979 012106 000773 BR 2$
3980 012110 012601 3$: MOV (SP)+,R1 ;RESTORE R1
3981 012112 006302 ASL R2
3982 012114 016200 013702' MOV FILTAB(R2),R0 ;SET POINTER
3983 012120 111402 MOVB (R4),R2 ;R2 GETS CHARACTER IN ASCII
3984 012122 020227 000101 CMP R2,#'A ;TEST FOR ASCII A
3985 012126 002406 BLT 5$ ;BR IF LESS THAN ASCII A
3986 012130 020227 000132 CMP R2,#'Z
3987 012134 003053 BGT 40$ ;BR IF GREATER THAN ASCII Z
3988 012136 162702 000100 SUB #100,R2 ;CHAR IS A THRU Z. SUBTRACT 100 TO CNVT TO RAD50
3989 012142 000416 BR 8$
3990
3991 012144 020227 000060 5$: CMP R2,#'0 ;TEST FOR ASCII 0 THRU 9
3992 012150 002406 BLT 6$ ;BR IF LESS THAN ASCII 0
3993 012152 020227 000071 CMP R2,#'9

```

```

3994 012156 003042          BGT      40$          ;BR IF > THAN ASCII 9
3995 012160 162702 000022    SUB      #22,R2      ;CHAR IS ASCII 0 THRU 9. SUB 22 TO CNVT TO RAD50
3996 012164 000405          BR        8$
3997
3998 012166 020227 000044    6$:      CMP      R2,#'$      ;TEST FOR A DOLLAR SIGN
3999 012172 001034          BNE      40$          ;BR IF NOT A DOLLAR
4000 012174 012702 000033    MOV      #33,R2      ;DOLLAR IS A 33
4001 012200 004767 000076    8$:      JSR      PC,60$      ;MULT CURRENT FILENAME WORD BY 50 AND THEN ADD R2
4002 012204 060210          ADD      R2,(R0)      ;ADD VALUE IN R2
4003 012206 005204          INC      R4          ;UPDATE INPUT STRING POINTER
4004 012210 000706          BR        10$          ;GET NEXT CHARACTER
4005
4006 012212 005204          9$:      INC      R4          ;UPDATE INPUT STRING POINTER
4007          ;WE ARRIVE HERE WHEN A PERIOD IS SPOTTED, OR WHEN
4008          ;THE INPUT ASCII STRING IS EMPTY
4009 012214          30$:      ;JUSTIFY LEFT AND ZERO FILL FILENAME OR EXTENSION.
4010 012214 012702 000006    MOV      #6,R2
4011 012220 160102          SUB      R1,R2          ;WE ARE COMPARING CONVERSION COUNT TO 6
4012 012222 001701          BEQ      10$          ;BR IF EAUAL TO 6.(NO JUSTIFY NEEDED)
4013 012224 002412          BLT      22$          ;BR IF WE ARE ON EXTENSION
4014 012226 020227 000003    CMP      R2,#3      ;SEE WHICH FILENAME WORD WE ARE ON
4015 012232 001003          BNE      20$          ;BR IF NOT ON WORD BOUNDARY
4016 012234 012701 000006    MOV      #6,R1          ;FUDGE CNTR TO ALLOW EXTENSION FILL
4017 012240 000672          BR        10$          ;GET EXTENSION CHARACTERS
4018 012242 004767 000034    20$:      JSR      PC,60$      ;MULT CURR&NT WORD BY 50
4019 012246 005201          INC      R1          ;INC THE CONVERSION COUNTER
4020 012250 000761          BR        30$          ;KEEP IT UP UNTIL WE REACH A BOUNDARY
4021
4022 012252 005402          22$:      NEG      R2          ;LEFT JUSTIFY THE EXTENSION
4023 012254 020227 000003    CMP      R2,#3
4024 012260 001662          BEQ      10$          ;BR IF EXTENSION ALREADY JUSTIFIED
4025 012262 002767          BLT      20$          ;BR TO JUSTIFY EXTENSION.
4026          ;ERROR. EXTENSION MORE THAN 3 CHARACTERS
4027 012264          40$:      TYPEMES #FLNMR,,CR ;ENTRY FOR FILENAME ERRORS
4028 012272 012604          MOV      (SP)+,R4      ;REPLACE INPUT STRING POINTER
4029 012274 022707          CMP      (PC)+,PC      ;SET C BIT, SKIP NEXT INSTRUCTION
4030 012276 005726          50$:      TST      (SP)+      ;DISCARD SAVED INPUT STRING POINTER
4031          ;CLC
4032 012300 000207          RTS      PC
4033
4034 012302          60$:      ;MULTIPLY (R0) BY 50
4035 012302 006310          ASL      (R0)
4036 012304 006310          ASL      (R0)
4037 012306 006310          ASL      (R0)          ;(R0) IS NOW MULTIPLIED BY 10 BASE 8
4038 012310 011046          MOV      (R0),-(SP)      ;SAVE (R0) MULTILPIED BY 10
4039 012312 006310          ASL      (R0)
4040 012314 006310          ASL      (R0)          ;(R0) NOW MULTIPLIED BY 40 BASE 8
4041 012316 062610          ADD      (SP)+,(R0)      ;10X +40X = 50X
4042 012320 000207          RTS      PC
4043
4044          .DSABL   LSB

```

```

4046 .SBTTL LOAD A FILE
4047
4048 .ENABL LSB
4049
4050 012322 DOLOAD: ;PERFORM A MAIN MEMORY OR WCS LOAD
4051 ;'EFFADR' HOLDS THE ADDRESS TO BEGIN LOADING AT
4052 ;
4053 ; IF LOADING WCS
4054 ; THEN IF MICRO CODE OPTION BITS ('MICOPT') ARE CLEAR
4055 ; THEN ONLY LOAD 2K MICRO WORDS
4056 ; ELSE LOAD ENTIRE FILE
4057 ;
4058 ;R0-->'TCONTL'
4059 ;R2-->'TSTRUN'
4060 012322 010005 MOV R0,R5 ;POINT R5 TO 'TCONTL'
4061 012324 012767 000002 010166 MOV #2,LNHCOD ;SET MICRO-CODE LENGTH TO LONG WORD
4062 012332 004712 JSR PC,(R2) ;TEST FOR STAR CPU RUNNING
4063 012334 103004 BCC 10$ ;BR IF NOT RUNNING
4064 012336 042767 000020 023256 5$: BIC #NOSHOW,FLAG ;CLEAR NOSHOW FLAG INCASE IT WAS SET
4065 012344 000207 RTS PC ;EXIT
4066
4067 012346 004767 176244 10$: JSR PC,TSTTY2 ;CLEAR CODE 2 MICRO-ERRORS
4068 012352 004767 176512 JSR PC,TSTCST ;TEST FOR CLOCK STOPPED
4069 012356 103767 BCS 5$ ;BR IF CLOCK IS STOPPED
4070 012360 112767 000002 023033 MOVB #LNLNH,CURLNH ;SET LOAD SIZE TO LONG WORD
4071 012366 005715 TST (R5) ;WCS LOAD?
4072 012370 100010 BPL 20$ ;BRANCH IF NO
4073 012372 032767 002000 023222 BIT #USEDEF,FLAG ;LOAD THE ECO FILE?
4074 012400 001404 BEQ 20$ ;BRANCH IF NO
4075 012402 012700 017202 MOV #ECONAM,R0 ;SETUP TO OPEN ECO FILE
4076 012406 004767 004526 JSR PC,SETFIL ;...
4077 012412 20$: OPEN$ #FILENM ;OPEN INPUT FILE
4078 012422 103745 BCS 5$ ;BR IF ERROR ON OPEN
4079 012424 105067 022772 CLRB CURADS ;ASSUME PHYSICAL LOAD
4080 ;CLC
4081 012430 112767 000001 010056 MOVB #1,LODFLG ;NOTE WE ARE LOADING A FILE (EDIT-21)
4082 012436 004767 174674 JSR PC,STCLMP ; CLEAR MEMORY MAPPING(DEPENDS ON C BIT)
4083 012442 005067 010100 CLR BYTSLD
4084 012446 005067 010076 CLR BYTSLD+2
4085 ;(SP) IS STARTING SECTOR
4086 ;2(SP) IS # OF SECTORS
4087 012452 012667 010060 MOV (SP)+,CURRSEC ;SAVE STARTING SECTOR
4088 012456 012667 010056 MOV (SP)+,SECSLF ;SAVE # OF SECTORS
4089 012462 001564 BEQ 50$ ;BR IF FILE EMPTY
4090 012464 005715 TST (R5) ;TEST FOR A WCS LOAD
4091 012466 100040 BPL 35$ ;BR IF NOT WCS LOAD
4092 012470 052714 000200 BIS #ROMNOP,(R4) ;SET ROM-NOP WHILE LOADING WCS
4093 012474 016700 024104 MOV EFFADR,R0 ;R0 GETS ADDRESS
4094 012500 005001 CLR R1 ;R1 IS UPPER ADDRESS BITS(ZERO)
4095 012502 012702 000042 MOV #WCSADD,R2 ;R2 IS ADDRESS OF 'WCSADD' ON IDBUS
4096 012506 004767 176244 JSR PC,WRITID ;WRITE TO 'WCSADD'
4097 012512 012737 000143 173030 MOV #WCSDAT!IDWRIT,&#IDCNTRL ;ADDRESS WCS DATA REG,ENABLE WRITE
4098 012520 000423 BR 35$
4099
4100 012522 005767 010016 3$: TST BYTSLF ;TEST FOR BLOCK BUFFER EMPTY

```


4101	012526	003074				BGT	44\$;BR IF NOT EMPTY
4102	012530	105767	022660			TSTB	ABORT		;ABORT SET VIA CONTROL-C?
4103	012534	001137				BNE	50\$;BR IF YES
4104	012536	005715				TST	(R5)		;WCS LOAD?
4105	012540	100013				BPL	35\$;BRANCH IF NO
4106	012542	105767	037744			TSTB	MICOPT		;MICRO CODE OPTIONS PRESENT?
4107	012546	001004				BNE	30\$;BRANCH IF YES (LOAD LIMIT IS 3K)
4108	012550	022767	060000	007770		CMP	#24576.,BYTSLD		;LOADED 2K MICRO WORDS YET?
4109	012556	101526				BLOS	50\$;BRANCH IF YES
4110	012560	022767	110000	007760	30\$:	CMP	#36864.,BYTSLD		;LOADED 3K MICRO WORDS YET?
4111	012566	101522				BLOS	50\$;BRANCH IF YES
4112	012570	005767	007744		35\$:	TST	SECSLF		;TEST FOR INPUT FILE EMPTY
4113	012574	003517				BLE	50\$;BR IF EMPTY
4114	012576				40\$:	F\$READ	CURRSEC,#USRBUF		;#<USRBSZ/128.>
4115									;FILL THE USER BUFFER
4116	012640	103004				BCC	42\$;BR IF NO ERRORS
4117	012642	012600				MOV	(SP)+,R0		;ERROR CODE TO R0
4118	012644	004767	000774			JSR	PC,TYFLER		;TYPE ERROR MSG AND CODE
4119	012650	000471				BR	50\$		
4120									
4121	012652	062767	000002	007656	42\$:	ADD	#<USRBSZ/128.>,CURRSEC		;UPDATE CURRENT SECTOR #
4122									;INIT BUFFER POINTER
4123	012660	012767	022600	007654		MOV	#USRBUF,BUFFRP		;SET BUFFER BYTE COUNT TO MAX
4124	012666	012767	000400	007650		MOV	#USRBSZ,BYTSLF		
4125	012674	162767	000002	007636		SUB	#<USRBSZ/128.>,SECSLF		;DECREMENT NUMBER OF SECTORS LEFT TO LOAD
4126									;BRANCH IF DONE OR MORE TO LOAD
4127	012702	002006				BGE	44\$;DECREMENT BUFFER BYTE COUNTER BY ONE SECTOR
4128	012704	162767	000200	007632	43\$:	SUB	#128.,BYTSLF		;ONLY LOAD WHAT WAS IN THE FILE
4129	012712	005267	007622			INC	SECSLF		
4130	012716	100772				BMI	43\$;... POINTS TO BUFFER
4131	012720	016701	007616		44\$:	MOV	BUFFRP,R1		
4132	012724	012100				MOV	(R1)+,R0		
4133	012726	012101				MOV	(R1)+,R1		
4134	012730	162767	000004	007606		SUB	#4,BYTSLF		;UPDATE BYTE COUNTER
4135	012736	062767	000004	007576		ADD	#4,BUFFRP		;UPDATE BUFFER POINTER
4136	012744	062767	000004	007574		ADD	#4,BYTSLD		;UPDATE NUMBER OF BYTES LOADED
4137	012752	005567	007572			ADC	BYTSLD+2		
4138	012756	005715				TST	(R5)		;DECIDE WHERE THESE 4 BYTES GO
4139	012760	100415				BMI	4\$;BR IF THEY GO TO WCS
4140	012762	010067	023574			MOV	R0,DATATO		;PUT THIS LONG WORD INTO STAR MAIN MEMORY
4141	012766	010167	023572			MOV	R1,DATATO+2		
4142	012772	004767	172414			JSR	PC,LOADDE		;DO A DEPOSIT
4143	012776	103416				BCS	50\$;BR TO EXIT IF ERROR ON DEPOSIT
4144	013000	062767	000004	023576		ADD	#4,EFFADR		;UPDATE ADDRESS
4145	013006	005567	023574			ADC	EFFADR+2		
4146	013012	000643				BR	3\$		
4147									
4148	013014	010037	173020		4\$:	MOV	R0,a#TOIDLO		;PUT DATA INTO 'TOID' REG
4149	013020	010137	173022			MOV	R1,a#TOIDHI		
4150	013024	052737	100000	173030		BIS	#IDCYCL,a#IDCNTL		;START THE WRITE
4151	013032	000633				BR	3\$		
4152									
4153	013034				50\$:				;REMEMBER 'ROMNOP' STILL SET IF WCS LOAD
4154									
4155	013034	004767	174342			JSR	PC,RESTMM		;RESTORE MEMORY MAPPING ENABLE

4156	013040					TYPES	#LOISDN,,CR	;TYPE "LOAD DONE"
4157	013046	012700	022546'			MOV	#BYTSLD,R0	;SET POINTER TO BYTE COUNTER
4158	013052	004767	171726			JSR	PC,R2GRAD	;R2 GETS CURRENT RADIX VALUE
4159	013056	005715				TST	(R5)	;LOADING WCS?
4160	013060	100411				BMI	51\$;BRANCH IF YES
4161	013062	012701	000004			MOV	#4,R1	;SET THE DATA LENGTH
4162	013066					CONVERT		;CONVERT STRING TO ASCII, RETURN PNTR IN R0
4163	013070					TYPES	R0	;TYPE # OF BYTES LOADED
4164	013074					TYPES	#BYTESL	;TYPE 'BYTES LOADED'
4165	013102	000444				BR	60\$;EXIT
4166								
4167								
4168								
4169	013104	005060	000002	51\$:		CLR	2(R0)	;SETUP TO CALCULATE # OF MICRO WORDS LOADED
4170	013110	162710	000014	52\$:		SUB	#12,,(R0)	;12 BYTES PER MICRO WORD
4171	013114	001404				BEQ	53\$;BRANCH IF DONE
4172	013116	103405				BLO	54\$;BRANCH IF DONE
4173	013120	005260	000002			INC	2(R0)	;UPDATE QUOTIENT
4174	013124	000771				BR	52\$	
4175	013126	005260	000002	53\$:		INC	2(R0)	;COUNT THE LAST DIVIDE
4176	013132	062700	000002	54\$:		ADD	#2,R0	;GET POINTER TO NUMBER OF MICRO WORDS LOADED
4177	013136	012701	000002			MOV	#2,R1	;SET THE DATA LENGTH
4178	013142					CONVERT		;CONVERT TO ASCII STRING
4179	013144					TYPES	R0	;TYPE NUMBER OF MICRO WORDS
4180	013150					TYPES	#MICWSL	;TYPE 'MICROWORDS LOADED'
4181	013156	004767	170470			JSR	PC,INITRT	;DO INIT ROUTINE
4182	013162	052767	004000	022432		BIS	#WCSPRES,FLAG	;REMEMBER WCS IS LOADED
4183	013170	032767	000020	022424		BIT	#NOSHOW,FLAG	;INHIBIT SHOWING VERSION?
4184	013176	001006				BNE	60\$;BRANCH IF YES
4185	013200	004767	176220			JSR	PC,GETVER	;GET PCS,WCS,FPLA VERSIONS
4186	013204	004767	173556			JSR	PC,DOSHR	;SHOW VERSIONS
4187	013210	004767	176110			JSR	PC,TSTVER	;CHECK FOR COMPATABILITY
4188	013214	042767	000020	022400	60\$:	BIC	#NOSHOW,FLAG	;CLEAR NOSHOW FLAG INCASE IT WAS SET
4189	013222	105067	007266			CLRB	LODFLG	; (EDIT-21)
4190	013226	000207				RTS	PC	
4191								
4192						.DSABL	LSB	
4193								
4194						.SBTTL	LINK COMMAND	
4195								
4196	013230					DOLINK:	;SET UP COMMAND LINKING	
4197	013230	012767	023200'	007320		MOV	#BUFO,INDBYT	;INIT BUFFER POINTER
4198	013236	012767	000016	007316		MOV	#14,,INDSEC	;INIT SECTOR PNTR TO LOGICAL SECTOR 14
4199	013244	012767	000012	007306		MOV	#10,,INDLFT	;MAX OF 10 SECTORS USED FOR LINKING
4200	013252	105267	007234			INCB	LINKG	;INITIATE LINKING
4201	013256	000207				RTS	PC	

```

4203 .SBTTL INDIRECT COMMAND LINE RETRIEVER
4204
4205 013260 INDLIN: ;ROUTINE TO GET A COMMAND LINE FROM A FLOPPY DISC FILE
4206 ;INPUTS: INDSEC = CURRENT LOGICAL SECTOR OF INDIRECT FILE
4207 ; INDBYT = BYTE POINTER INTO CURRENT LOGICAL SECTOR
4208 ; INDLFT = NUMBER OF SECTORS LEFT IN FILE
4209 ;OUTPUTS: C BIT SET IF FILE EMPTY OR FLOPPY ERROR
4210 ; IF <C BIT CLEAR> THEN <'TTYBUF' CONTAINS A COMMAND LINE>
4211 ;EFFECTS: IF <END-OF-FILE DETECTED> THEN <"aEOF" IS PRINTED ON TERMINAL>
4212
4213 013260 012703 035421' MOV #TTYBUF+1,R3 ;POINT R3 TO COMMAND LINE
4214 013264 010301 MOV R3,R1 ;DITTO R1
4215 013266 016702 007264 MOV INDBYT,R2 ;POINT R2 TO FLOPPY BUFFER
4216 013272 020227 023400' 5$: CMP R2,#BUF0+128. ;TEST FOR CURRENT FLOPPY SECTOR DONE WITH
4217 013276 002453 BLT 40$ ;BR IF MORE CHARACTERS IN CURRENT BUFFER
4218 013300 005367 007254 DEC INDLFT ;MINUS ONE FROM # OF SECTORS LEFT
4219 013304 002011 BGE 20$ ;BR IF ONE OR MORE SECTORS ARE LEFT
4220 013306 012701 022405' 10$: MOV #EOFMES,R1 ;R1 GETS PNTR TO '<aEOF>' MESSAGE
4221 013312 004767 000166 JSR PC,INDECH ;TYPE IF NOT BOOTING
4222 013316 042767 000200 022276 11$: BIC #INDMOD,FLAG ;DISABLE INDIRECT COMMAND MODE
4223 013324 000261 SEC ;C BIT SET INDICATES FAILURE
4224 013326 000465 BR 80$
4225
4226 013330 012702 023200' 20$: MOV #BUF0,R2 ;R2 GETS BUFFER PNTR
4227 013334 026767 007222 007222 CMP INDLFT,SECLD ;SECTOR WE WANT ALREADY LOADED?
4228 013342 001424 BEQ 25$ ;BR AND SKIP READ IF YES
4229 013344 F$READ INDLFT,R2 ;READ NEXT SECTOR AND WAIT FOR COMPLETION OF READ
4230 013402 103004 BCC 25$ ;BR IF NO ERROR ON READ
4231 013404 012600 MOV (SP)+,R0 ;R0 GETS ERROR CODE
4232 013406 004767 000232 JSR PC,TYPEFLR ;TYPE ERROR MSG AND CODE
4233 013412 000741 BR 11$
4234
4235 013414 016767 007142 007142 25$: MOV INDLFT,SECLD ;REMEMBER SECTOR WE LOADED
4236 013422 005267 007134 INC INDLFT ;UPDATE CURRENT SECTOR #
4237 013426 112211 40$: MOVB (R2)+,(R1) ;MOVE A CHARACTER FROM FLOPPY BUFFER TO TERMINAL BUFFER
4238 013430 001726 BEQ 10$ ;BR IF CHARACTER IS BLANK
4239 013432 020127 036541' CMP R1,#TTYBUF+81. ;CHECK FOR TTY BUFFER OVERFLOW
4240 013436 002404 BLT 50$ ;BR IF NOT OVERFLOWING BUFFER
4241 013440 49$: TYPEMES #BADLIN,,CR ;ERROR. TTY BUFFER OVERFLOW OR UNDERFLOW
4242 013446 000723 BR 11$
4243
4244 013450 122127 000012 50$: CMPB (R1)+,#12 ;CHECK FOR END-OF-LINE(LINE FEED CHARACTER)
4245 013454 001306 BNE 5$ ;BR IF NOT END-OF-LINE
4246 013456 010267 007074 MOV R2,INDBYT ;END-OF-LINE. SAVE CURRENT BUFFER POINTER.
4247 013462 160301 SUB R3,R1 ;COMPUTE NUMBER OF CHARACTERS IN LINE
4248 013464 003765 BLE 49$ ;BR IF LINE SIZE GOES TO ZERO OR NEGATIVE
4249 013466 110143 MOVB R1,-(R3) ;SAVE LENGTH OF LINE IN TTY BUFFER
4250 013470 010301 MOV R3,R1 ;R1 GETS PNTR TO LINE FOR ECHO PURPOSES
4251 013472 004767 000006 JSR PC,INDECH ;ECHO IF NOT BOOTING
4252 013476 105311 DECB (R1) ;GET RID OF LINEFEED AT END OF LINE
4253 013500 000241 CLC ;CLEAR C BIT TO INDICATE LINE WAS RETRIEVED SUCCESSFULLY.
4254 013502 000207 80$: RTS PC
4255
4256 013504 INDECH: ;CONDITIONAL PRINTER FOR INDIRECT COMMAND FILE PROCESSING
4257 ;IF<NOT BOOTING> THEN <TYPE MESSAGE WHOSE ADDRESS IS IN R1>

```

72-ESKAA-10.1 INDIRECT COMMAND LINE RETRIEVER
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 51-1
INDIRECT COMMAND LINE RETRIEVER

B 8

20-MAY-1986

Fiche 1 Frame B8

Sequence 92

4258 013504 105767 007001
4259 013510 001002
4260 013512
4261 013516 000207

90\$:

TSTB NOECHO
BNE 90\$
TYPEMES R1,,CR
RTS PC

;ECHO SUPPRESSED?
;BK IF YES

ZZ-ESKAA-10.1 OPEN FILE,TYPE FLOPPY ERROR MESSAGE
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 52
 OPEN FILE,TYPE FLOPPY ERROR MESSAGE

```

4263 .SBTTL OPEN FILE,TYPE FLOPPY ERROR MESSAGE
4264
4265 013520 OPENER: ;ROUTINE TO OPEN A FLOPPY FILE ON DRIVE 0 OR 1
4266 ;INPUTS: 'DX1FLG' = 1 IF DRIVE 1 IS TO BE USED
4267 ; FILENAME POINTER ON STACK AT 2 (SP)
4268 ;OUTPUTS: C BIT SET IF OPEN FAILS(ERROR IS PRINTED)
4269 ; C BIT CLEAR IF OPEN OK, AND (SP) = STARTING SECTOR
4270 ; 2(SP) = # OF SECTORS IN FILE
4271 013520 016667 000002 007026 MOV 2(SP),FILPNT ;SET FILE NAME POINTER
4272 013526 032767 000040 021644 BIT #DX1FLG,TCONTL ;DETERMINE PROPER DRIVE TO USE
4273 013534 001014 BNE 15$ ;BR IF DRIVE 1
4274 013536 F$OPEN FILPNT ;OPEN FILE ON DRIVE 0
4275 013544 103014 BCC 20$ ;BR IF FILE FOUND
4276 013546 012600 5$: MOV (SP)+,R0 ;ERROR CODE TO R0
4277 013550 004767 000044 JSR PC,TPERRM ;TYPE ERROR MESSAGE
4278 013554 011666 000002 MOV (SP),2(SP) ;CLEAR STACK
4279 013560 005726 TST (SP)+
4280 013562 000261 SEC ;C BIT SET INDICATES OPEN FAILED
4281 013564 000414 BR 30$
4282
4283 013566 15$: F$OPN1 FILPNT ;OPEN FILE ON DRIVE 1
4284 013574 103764 BCS 5$ ;BR IF ERROR ON OPEN
4285 013576 016666 000002 000006 20$: MOV 2(SP),6(SP) ;RETURN STARTING SECTOR AND # OF SECTORS ON STACK
4286 013604 016666 000004 000002 MOV 4(SP),2(SP)
4287 013612 012666 000002 MOV (SP)+,2(SP)
4288 013616 000207 30$: RTS PC
4289
4290 013620 TPERRM: ;TYPE ERROR MESSAGE.
4291 013620 020027 000002 CMP R0,$$FNF ;FILE NOT FOUND?
4292 013624 001004 BNE 10$ ;BR IF NO
4293 013626 TYPEMES #NOSUFL,,CR ;TYPE 'FILE NOT FOUND'
4294 013634 000402 BR 20$
4295
4296 013636 004767 000002 10$: JSR PC,TYFLER ;TYPE ERROR MESSAGE AND CODE
4297 013642 000207 20$: RTS PC
4298
4299 013644 TYFLEP: ;ROUTINE TO TYPE FLOPPY ERROR MESSAGE AND CODE #
4300 ;R0=CODE # OF ERROR
4301 013644 TYPEMES #FLPERR,,CR ;TYPE <CRLF><TAB>?FLOPPY ERR,CODE=)
4302 013652 TYPERC: ;ROUTINE TO TYPE THE ERROR CODE IN R0
4303 013652 110067 006631 MOVB R0,ERRCCD
4304 013656 012746 022507' MOV #ERRCCD,-(SP)
4305 013662 012746 000001 MOV #1,-(SP)
4306 013666 012746 000020' MOV #16,-(SP)
4307 013672 004767 140022' JSR PC,CONVRT
4308 013676 TYPEMES
4309 013700 000207 RTS PC
4310
4311 ;POINTER ARRAY
4312 013702 022546' FILTAB: .WORD FILENM
4313 013704 022550' .WORD FILENM+2
4314 013706 022552' .WORD EXTENS

```

4316 .SBTTL TIMEOUT/ODD ADDRESS TRAP CATCHER

4317

4318 013710 012706 001000

4319 013714 012746 123456

4320 013720

4321 013734 000167 165274

4322

ODDADD: MOV #1000,SP ;RESET STACK POINTER
MOV #123456,-(SP) ;THIS WILL PREVENT INIT AND WCS ECO LOAD
T\$WRIT #TIMTRP,#TIMEND-TIMTRP ;PRINT ?TRAP-4,RESTARTING CONSOLE
JMP RESTRT ;RESTART CONSOLE PROGRAM

```

4331
4332
4333
4334
4335
4336
4337
4338
4339
4340 013740 005721      ENLOCN: TST      (R1)+      ;ENABLE LOCAL CONTROL
4341 013742 005721      ENLOCJ: TST      (R1)+      ;ENABLE LOCAL COPY
4342 013744 005721      ENECHO: TST      (R1)+      ;ENABLE REMOTE ECHO
4343 013746          DSCLER: ;ENTRY FOR DISABLE CARRIER LOSS ERROR MESSAGE
4344 013746 132767 000002 037750' BITB      #REMOT,LASPOS ;IN REMOTE MODE?
4345 013754 001403          BEQ      10$          ;BR IF NOT, AND NULLIFY COMMAND
4346 013756 056167 014012' 021546 BIS      BITTAB(R1),TCTFLG ;SET APPROPRIATE BIT OF TERMINAL CONTROL FLAG
4347 013764 000207          10$: RTS      PC
4348
4349 013766 005721      DSLOCJ: TST      (R1)+      ;DISABLE LOCAL COPY
4350 013770 005721      DSECHO: TST      (R1)+      ;DISABLE REMOTE ECHO
4351 013772          ENCLER: ;ENTRY FOR ENABLE CARRIER ERROR MESSAGE
4352 013772 132767 000002 037750' BITB      #REMOT,LASPOS ;IN REMOTE MODE?
4353 014000 001403          BEQ      10$          ;BR IF NOT, NULLIFYING COMMAND
4354 014002 046167 014012' 021522 BIC      BITTAB(R1),TCTFLG ;CLEAR PROPER BIT OF TERMINAL CONTROL FLAG
4355 014010 000207          10$: RTS      PC
4356
4357 014012          BITTAB: ;TABLE OF BIT VALUES FOR FLAGS IN 'TCTFLG'(TERMINAL CONTROL FLAG)
4358 014012 002000          .WORD     DISCAR      ;DISABLE CARRIER LOSS ERROR BIT
4359 014014 001000          .WORD     REMECH      ;REMOTE ECHO ENABLE BIT
4360 014016 000000          .WORD     LOCCOP      ;LOCAL COPY ENABLE BIT
4361 014020 000000          .WORD     LOCCNT!LOCCOP ;ENABLE LOCAL CONTROL BIT & LOCAL COPY
4362
4363          .ENABL  LSB
4364 014022          DSFLOP: ;ROUTINE TO ENABLE/DISABLE LOCAL FLOPPY DRIVE
4365 014022 105767 022572      TSTB      ALLCOP      ;REMOTE DISABLED ALREADY?
4366 014026 001013          BNE      20$          ;TYPE ERROR
4367 014030 005201          INC      R1          ;ENTRY TO DISABLE LOCAL FLOPPY
4368 014032 110167 022136      ENFLOP: MOVB      R1,ALLREM ;ENTRY TO ENABLE LOCAL FLOPPY DRIVE
4369 014036 000207          RTS      PC
4370
4371          DSREMT: ;ROUTINE TO ENABLE/DISABLE REMOTE 'FLOPPY'
4372 014040 105767 022130      TSTB      ALLREM      ;LOCAL DISABLED ALREADY?
4373 014044 001004          BNE      20$          ;TYPE ERROR
4374 014046 005201          INC      R1          ;MARK FOR DISABLE
4375 014050 110167 022544      ENREMT: MOVB      R1,ALLOC ;SET ENABLE/DISABLE FLAG
4376 014054 000207          RTS      PC
4377 014056          20$: TYPMES  #DISERR,,CR      ;TYPE ERROR MESSAGE
4378 014064          25$: TYPMES  #CANTDO      ;TYPE 'FUNCTION ABORTED'
4379 014072 000207          RTS      PC
4380

```

4382 014074

DOXLOA: .SBTTL APT 'X' COMMAND EXECUTION

4383

4384

4385

4386

4387

4388

4389

4390

4391

4392

4393

4394

4395

4396

4397

4398

4399

4400

4401

4402

4403

4404

4405

4406 014074 105767 037747'

4407 014100 001004

4408 014102

4409 014110 000765

4410

4411 014112 004767 175006

4412 014116 103476

4413 014120 105077 037766'

4414 014124 012701 036420'

4415 014130 112100

4416 014132 005003

4417

4418 014134 112102

4419 014136 060203

4420 014140 105300

4421 014142 002374

4422

4423 014144 016701 037766'

4424

4425 014150 116702 022606

4426 014154 060203

4427 014156 105703

4428 014160 001047

4429

4430 014162 012746 022414'

4431 014166

4432 014170 005003

4433 014172 112767 000001 021212

4434 014200 105067 021215

4435

4436 014204 105711

;THIS ROUTINE PERFORMS A BINARY DUMP TO 11/780 MEMORY
;REFER TO MID RANGE CONSOLE SPEC. FOR 'X' COMMAND DETAILS

;THE COMMAND IS ONLY VALID IF THE PROGRAM WAS LOADED BY APT
;AND THE CPU IS HALTED. DUE TO THE SLOWNESS OF COMMAND PARSING
;THE COMMAND CHECKSUM WILL BE RECEIVED (AND THEREFORE BE PROCESSED
;BY THE REMOTE INTERRUPT SERVICE ROUTINE) BEFORE THIS ROUTINE IS
;CALLED. THEREFORE, THE INTERRUPT SERVICE ROUTINE PLACES ALL
;CHARACTERS IN LOCATION 'XCMSV' AND BY THE TIME THIS ROUTINE IS
;CALLED, THE COMMAND CHECKSUM WILL BE THERE.

;ALSO, ALL ERROR MESSAGES FROM THE MEMORY DEPOSIT ROUTINE ARE
;SUPPRESSED SINCE APT WOULDN'T KNOW WHAT TO DO WITH THEM. THEREFORE,
;THERE IS NO CHECK THAT THE DATA WAS DEPOSITED CORRECTLY.

; IMPLICIT INPUTS:

; COUNT - CONTAINS THE NUMBER OF BYTES TO DEPOSIT
; EFFADR- CONTAINS THE START ADDRESS OF THE ADDRESS
; XCMSV- CONTAINS THE RECEIVED CHECKSUM OF THE COMMAND

TSTB APTLOD ;DID APT LOAD US ?
BNE 30\$;BR IF SO
TYPEMES #NOTREM,,CR ;NO REMOTE ACCESS
BR 25\$;FINISH ERROR MSG AND EXIT

30\$: JSR PC,TSTRUN ;CPU RUNNING?
BCS 40\$;EXIT, IF SO
CLRB #RMRCR ;CLEAR RECEIVER INT. ENABLE
MOV #TTYBUF,R1 ;R1 GETS INPUT LINE ADDRESS
MOVB (R1)+,R0 ;R0 GETS NUMBER OF BYTES IN BUFFER(EDIT 4-05)
CLR R3

5\$: MOVB (R1)+,R2 ;GET CHAR
ADD R2,R3 ;ADDITIVE CHECKSUM
DECB R0 ; DECREMENT THE BYTE COUNT
BGE 5\$;END OF BUFFER? COUNT THE CARRIAGE RETURN ALSO
;EDIT 4-05
MOV RMRCR,R1 ;R1 GETS TERMINAL BUFFER ADDRESS

45\$: MOVB XCMSV,R2 ; GET THE CHECKSUM CHARACTER
ADD R2,R3 ; ADD TO COMMAND CHECKSUM
TSTB R3
BNE 90\$;COMMAND CHECKSUM ERROR

10\$: MOV #CONPMP,-(SP) ;TYPE CONSOLE PROMPT AS 'ACK'
TYPEMES
CLR R3 ;R3 IS NEW CHECKSUM
MOVB #1,DEEXBY ;SET FOR DEPOSIT(NON-ZERO)
CLRB CURLNH ;SET CURRENT DATA LENGTH AS 1 BYTE

15\$: TSTB (R1) ;GOT DATA CHAR YET?

ZZ-ESKAA-10.1

APT 'X' COMMAND EXECUTION

V10-01-L

MACRO V05.03 Friday 25-Apr-86 10:56 Page 55-1

APT 'X' COMMAND

EXECUTION

```

4437 014206 100376          BPL      15$
4438
4439 014210 116167 000002 022344  MOVB   2(R1),DATATO ;GET BYTE TO DEPOSIT FROM RBUF
4440 014216 066703 022340      ADD   DATATO,R3 ;ADDITIVE CHECKSUM
4441 014222 005367 021160      DEC   COUNT
4442 014226 100417          BMI     35$ ;MORE TO DEPOSIT?
4443 014230 010146          MOV     R1,-(SP) ;SAVE R1
4444 014232 005067 021146      CLR   NEXTCT ;SET FOR ONE DEPOSIT ONLY (NO 'NEXT')
4445 014236 012746 014250'    MOV     #50$,-(SP) ; PUSH RETURN PC
4446 014242 004067 171174      JSR    RO,MICAST ; DO THE DEPOSIT
4447 014246 000440          .WORD   CPHYSE ; POINTER TO MICROCODE ROUTINE
4448 014250 062767 000001 022326 50$: ADD     #1,EFFADR ; INCREMENT THE ADDRESS
4449 014256 005567 022323      ADC   EFFADR+1
4450 014262 012601          MOV     (SP)+,R1 ;RESTORE R1
4451 014264 000747          BR      15$ ;IGNORE ANY ERRORS
4452
4453 014266 105703          35$: TSTB   RS
4454 014270 001411          BEQ     40$ ;CHECKSUM VALID?
4455 014272 012746 022157'    MOV     #XERR2,-(SP) ;DATA CHECKSUM ERROR
4456 014276 000402          BR      95$
4457
4458 014300 012746 022146'    90$: MOV     #XERR1,-(SP) ;COMMAND CHECKSUM ERROR
4459 014304          95$: TYPEMES
4460 014306          TYPEMES #XERR3
4461 014314 052711 000100    40$: BIS     #RCVINT,(R1)
4462 014320 105067 022435      CLRB   XLOFLG
4463 014324 000207          RTS     PC
4464
4465          .DSABL   LSB
4466
4467          000366      APTRTN=.-ENLOCN ;USED TO COMPUTE SIZE OF APT RELATED FUNCTIONS
4468

```

```
4470 .SBTTL
4471 .SBTTL PARSING TABLES AND ACTIONS
4472
4499
4500
4544
4545 ;NODE OFFSET DEFINITIONS
4546
4547 ;INFO=0
4548 000002 ACTION=2
4549 000004 YESLINK=4
4550 000005 NOLINK=5
4551 000006 MNOSIZ=6 ;EACH NODE IS 6 BYTES LONG
```

```
4553 .SBTTL
4554 .SBTTL  PARSE
4555
4556 014326 RECOG: ;SEE IF AN ASCII STRING CAN BE RECOGNIZED AS A COMMAND
4557 ;INPUTS: R3-->A NODE OF A SYNTAX CHECK TREE
4558 ; R4-->ASCII STRING
4559 ; R5-->ROOT NODE OF SYNTAX TREE BEING USED
4560 ; R0,R1,R2 ARE SCRATCH PADS
4561 ;OUTPUTS: C BIT SET IF ASCII STRING IS NOT A RECOGNIZED COMMAND
4562
4563 ;EFFECTS: IF THE NEXT LEXEME IN THE INPUT STRING IS EQUIVALENT
4564 ; TO THE STRING POINTED TO BY INFO(R3), THE ACTION ASSOCIATED WITH
4565 ; THE CURRENT NODE IS PERFORMED, AND R3 IS UPDATED
4566 ; TO POINT TO THE NEXT NODE VIA THE YESLINK(R3).
4567 ; R4 IS UPDATED TO POINT TO THE NEXT CHARACTER IN
4568 ; THE INPUT STRING PAST THE PART OF THE STRING THAT
4569 ; WAS RECOGNIZED.
4570
4571 ; IF THE NEXT LEXEME IN THE INPUT STRING IS NOT EQUIVALENT
4572 ; TO THE STRING POINTED TO BY INFO(R3), R3 IS UPDATED TO
4573 ; POINT TO THE NEXT NODE VIA THE NOLINK(R3).
4574 ; R4 IS UNCHANGED.
4575
4576 ; IN EITHER CASE, THIS PROCESS CONTINUES UNTIL R3=0.
4577
4578 ;IF <R3=0 AND RECOGNITION FAILED> THEN <RETURN WITH C BIT SET>
4579 ; ELSE <RETURN WITH C CLEAR>
4580
4581 ;SPECIAL EFFECTS:
4582 ; IF THE NEXT LEXEME IN THE INPUT STRING IS A QUALIFIER, THE CURRENT NODE
4583 ; POINTER IS SAVED, AND REPLACED BY A POINTER TO THE ROOT OF THE QUALIFIER
4584 ; TREE. THEN THIS ROUTINE CALLS ITSELF TO PROCESS THE QUALIFIER.
4585 .ENABL  LSB
4586
4587 014326 004767 000272 1$: JSR PC,REMLEA ;THROW AWAY LEADING BLANKS IN THE INPUT STRING
4588 014332 103003 BCC 4$ ;BR IF NO LEADING BLANKS
4589 014334 020527 016410' CMP R5,#QALTRE ;SEE IF WE ARE PROCESSING A QUALIFIER
4590 014340 001525 BEQ 60$ ;BR IF WE ARE(A BLANK IS END OF A QUALIFIER)
4591 014342 121427 600057 4$: CMPB (R4),# '/' ;TEST FOR A SLASH IN INPUT STRING
4592 014346 001021 BNE 5$ ;BR IF NOT A SLASH
4593 014350 005204 INC R4 ;POSITION INPUT STRING POINTER PAST THE SLASH
4594 014352 020527 016410' CMP R5,#QALTRE ;SEE IF WE ARE ALREADY PROCESSING A QUALIFIER
4595 014356 001002 BNE 2$ ;BR IF NOT
4596 014360 010503 MOV R5,R3
4597 014362 000761 BR 1$
4598
4599 014364 010346 2$: MOV R3,-(SP) ;SAVE CURRENT NODE POINTER
4600 014366 010546 MOV R5,-(SP) ;AND TREE ROOT POINTER
4601 014370 012705 016410' MOV #QALTRE,R5 ;R5 GETS POINTER TO ROOT OF QUALIFIER TREE
4602 014374 010503 MOV R5,R3 ;R3 LIKEWISE
4603 014376 004767 177724 JSR PC,RECOG ;RECOGNIZE THE QUALIFIER
4604 014402 012605 MOV (SP)+,R5
4605 014404 012603 MOV (SP)+,R3 ;RESTORE POINTERS
4606 014406 103505 BCS NULL ;BR IF ERROR ON QUALIFIER
4607 014410 000746 BR 1$ ;CONTINUE IN MAINTREE
```

4608							
4609	014412	105763	000005	5\$:	TSTB	NOLINK(R3)	;SEE IF RECOG NUMBER OR ASCII STRING TO RECOGNIZE
4610	014416	100443			BMI	50\$;BR IF NUMBER TO RECOGNIZE
4611	014420	105763	000004		TSTB	YESLINK(R3)	;SEE IF ONE STRING OR LIST OF STRINGS TO CHECK
4612	014424	100011			BPL	7\$;BR IF JUST ONE STRING
4613	014426	011346			MOV	(R3),-(SP)	;PROCESS A LIST(SAVE POINTER)
4614	014430	017602	000000	6\$:	MOV	a(SP),R2	;R2 GETS POINTER TO A CHECK STRING
4615	014434	001002			BNE	61\$;BR IF MORE TO CHECK
4616	014436	005726			TST	(SP)+	;REMOVE POINTER FROM STACK
4617	014440	000412			BR	10\$	
4618							
4619	014442	062716	000002	61\$:	ADD	#2,(SP)	;POINT TO NEXT STRING POINTER FOR NEXT PASS
4620	014446	000401			BR	8\$	
4621							
4622	014450	011302		7\$:	MOV	(R3),R2	;R2 GETS POINTER TO CHECK STRING
4623	014452	004767	000232	8\$:	JSR	PC,RECSTR	;CHECK INPUT STRING AGAINST CHECK STRING
4624	014456	103011			BCC	20\$;BR IF STRING WAS RECOGNIZED
4625	014460	105763	000004		TSTB	YESLINK(R3)	;ARE WE PROCESSING A LIST OF STRINGS?
4626	014464	100761			BMI	6\$;BR IF WE ARE
4627	014466	116300	000005	10\$:	MOVB	NOLINK(R3),R0	
4628	014472	004767	000152		JSR	PC,COMPX	;SET R3 TO ADDRESS OF NEXT NODE VIA 'NOLINK'
4629	014476	103713			BCS	1\$;BR IF NOT AT A TREE FRONTIER
4630	014500	000447			BR	70\$	
4631							
4632	014502	105763	000004	20\$:	TSTB	YESLINK(R3)	;ARE WE PROCESSING A LIST?
4633	014506	100021			BPL	23\$;BR IF WE ARE NOT
4634	014510	161316			SUB	(R3),(SP)	
4635	014512	162716	000002		SUB	#2,(SP)	
4636	014516	066316	000002		ADD	ACTION(R3),(SP)	
4637	014522	013600			MOV	a(SP)+,R0	;R0 GETS ACTION RTN POINTER
4638	014524	000414			BR	24\$	
4639							
4640	014526	105763	000004	50\$:	TSTB	YESLINK(R3)	;CHECK FOR A SPECIAL TEST
4641	014532	100003			BPL	55\$;BR IF NOT A SPECIAL TEST
4642	014534	004773	000000		JSR	PC,a(R3)	;PERFORM A SPECIAL TEST ROUTINE
4643	014540	000402			BR	59\$	
4644							
4645	014542	004767	000340	55\$:	JSR	PC,RECNUM	;TRY TO RECOGNIZE A NUMBER
4646	014546	103016		59\$:	BCC	25\$;BR IF RECOGNIZED
4647	014550	000746			BR	10\$	
4648							
4649	014552	016300	000002	23\$:	MOV	ACTION(R3),R0	;R0 GETS ACTION ROUTINE POINTER
4650	014556	012702	035416	24\$:	MOV	#WHATTODO,R2	;R2 GETS HANDY POINTER FOR ACTION ROUTINES
4651	014562	005001			CLR	R1	;R1 CLEAR IS REQUIRED FOR SOME ACTIONS
4652					;CLC		;C BIT CLEAR ALSO REQUIRED FOR SOME ACTIONS
4653	014564	032700	000001		BIT	#1,R0	;ACTION POINTER 'ODD'?
4654	014570	001404			BEQ	26\$;BR IF NOT
4655	014572	042700	000001		BIC	#1,R0	;MAKE POINTER EVEN
4656	014576	010012			MOV	R0,(R2)	;SAVE ROUTINE POINTER
4657	014600	000401			BR	25\$	
4658							
4659	014602	004710		26\$:	JSR	PC,(R0)	;DO THE ACTION ROUTINE
4660	014604	116300	000004	25\$:	MOVB	YESLINK(R3),R0	;UPDATE CURRENT NODE POINTER VIA YESLINK
4661	014610	004767	000044		JSR	PC,COMPX	;R3 GETS ADDRESS OF NEXT NODE
4662	014614	103644			BCS	1\$;BR IF NOT AT TREE FRONTIER

ZZ-ESKAA-10.1 PARSE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:54 Page 57-2
PARSER

K 8

20-MAY-1986

Fiche 1 Frame K8

Sequence 101

4663 014616 005727
4664 014620 000261
4665 014622 000207
4666
4667

60\$: TST (PC)+
70\$: SEC
NULL: RTS PC
.DSABL LSB

;CLEAR C BIT

```

4669 .SBTTL REMOVE BLANKS, COMPUTE NEXT NODE ADDRESS
4670
4671 014624 REMLEA: ;REMOVE LEADING BLANKS FROM A STRING
4672 ;R4-->STRING
4673 ;OUTPUTS: R4-->FIRST CHARACTER NOT A BLANK(SPACE OR TAB)
4674 ; C BIT SET IF A LEAST ONE LEADING BLANK REMOVED
4675 014624 005046 CLR -(SP) ;CREATE A FLAG FOR LEADING BLANKS
4676 014626 121427 000040 10$: CMPB (R4), #' ;CHECK FOR A SPACE
4677 014632 001003 BNE 30$ ;BR IF NOT A SPACE
4678 014634 005204 20$: INC R4
4679 014636 005216 INC (SP) ;REMEMBER A BLANK SEEN
4680 014640 000772 BR 10$
4681
4682 014642 121427 000011 30$: CMPB (R4), #11 ;CHECK FOR A TAB
4683 014646 001772 BEQ 20$ ;BR IF A TAB
4684 014650 005726 TST (SP)+ ;CHECK FOR FLAG SET
4685 ;CLC
4686 014652 001401 BEQ 35$ ;BR IF NO LEADING BLANKS REMOVED
4687 014654 000261 SEC
4688 014656 000207 35$: RTS PC
4689
4690 014660 COMPNX: ;COMPUTE ADDRESS OF NEXT NODE OF CURRENT TREE
4691 ;R0=NODE # OF NEXT NODE(0 IF TREE FRONTIER REACHED)
4692 ;R5=POINTER TO ROGR OF CURRENT TREE
4693 ;OUTPUTS:
4694 ; C BIT CLEAR IF TREE FRONTIER REACHED, AND R3 IS CLEAR
4695 ; ELSE: R3 POINTS TO NEXT TREE NODE
4696 ;
4697 ; R0 AND R1 ARE NOT PRESERVED
4698
4699 014660 005003 CLR R3 ;CLEAR CURRENT NODE POINTER
4700 ;CLC
4701 014662 042700 177600 BIC #177600, R0 ;CLEAR UPPER BIT OF LOWER BYTE OF R0
4702 014666 001407 BEQ 50$ ;BR IF TREE FRONTIER REACHED
4703 014670 012701 000006 40$: MOV #MNOSIZ, R1 ;R1 GETS THE SIZE OF A NODE IN BYTES
4704 014674 060003 ADD R0, R3 ;R3 GETS NODE #(R0) TIMES NODE SIZE(R1)
4705 014676 005301 DEC R1
4706 014700 003375 BGT 40$
4707 014702 060503 ADD R5, R3 ;ADD IN TREE ROOT ADDRESS
4708 014704 000261 SEC
4709 014706 000207 50$: RTS PC

```

```

4711 .SBTTL RECOGNIZE A STRING OF ASCII CHARACTERS
4712
4713 .ENABL LSB
4714
4715 014710 RECSTR: ;R4-->INPUT STRING
4716 ;R2-->CHECK STRING
4717 ;OUTPUTS:
4718 ; C BIT CLEAR IF STRING RECOGNIZED, R4 POINTS TO NEXT PART OF STRING.
4719 ; C BIT SET IF STRING NOT RECOGNIZED, R4 UNCHANGED
4720 014710 010446 MOV R4, -(SP)
4721 014712 004767 000070 JSR PC, TESTND ;TEST FOR A DELIMITER
4722 014716 103407 BCS 30$ ;BR IF IT WAS NOT A DELIMITER
4723 014720 121412 CMPB (R4), (R2) ;SEE IF CHECK STRING WAS LOOKING FOR A DELIMITER
4724 014722 001025 BNE NOMATC ;BR IF IT WAS NOT(TAKE NO MATCH EXIT)
4725 014724 005204 INC R4 ;UPDATE INPUT STRING POINTER PAST DELIMITER
4726 014726 000425 BR MATCH ;TAKE MATCH EXIT
4727
4728 014730 120114 10$: CMPB R1, (R4) ;COMPARE INPUT TO CHECK STRING
4729 014732 001021 BNE NOMATC ;BR IF THEY ARE DIFFERENT(NO MATCH)
4730 014734 122224 20$: CMPB (R2)+, (R4)+ ;ADD 1 TO BOTH STRING POINTERS
4731 014736 111201 30$: MOVB (R2), R1 ;CHECK FOR FIRST 'FENCE' IN CHECK STRING.
4732 014740 100373 BPL 10$ ;BR IF NOT AT FIRST FENCE YET
4733 014742 112201 40$: MOVB (R2)+, R1 ;TEST FOR SECOND FENCE IN CHECK STRING
4734 014744 100405 BMI 60$ ;BR IF NOT AT SECOND FENCE
4735 014746 004767 000034 50$: JSR PC, TESTND ;CHECK FOR A DELIMITER IN INPUT STRING
4736 014752 103013 BCC MATCH ;BR TO MATCH EXIT IF DELIMITER SEEN
4737 014754 005204 INC R4 ;UPDATE INPUT STRING POINTER
4738 014756 000773 BR 50$ ;KEEP IT UP UNTIL INPUT STRING POINTER TO NEXT DELIMITER
4739
4740 014760 004767 000022 60$: JSR PC, TESTND ;CHECK FOR A DELIMITER IN INPUT STRING
4741 014764 103006 BCC MATCH ;BR IF DELIMITER TO MATCH EXIT
4742 014766 042701 177600 BIC #177600, R1 ;CLEAR UPPER BITS OF R1
4743 014772 120124 CMPB R1, (R4)+ ;CHECK INPUT AGAINST CHECK STRING
4744 014774 001762 BEQ 40$ ;BR IF EQUAL, KEEP CHECKING
4745 014776 012604 NOMATC: MOV (SP)+, R4 ;NO MATCH. RESTORE INPUT STRING POINTER
4746 015000 022707 CMP (PC)+, PC ;SET C BIT, AND SKIP NEXT INSTRUCTION
4747 015002 005726 MATCH: TST (SP)+ ;MATCH EXIT. DISCARD SAVED POINTER, CLEAR C BIT
4748 ;CLC
4749 015004 000207 RTS PC
4750
4751 .DSABL LSB

```

```

4753 .SBTTL CHECK FOR A DELIMITER IN INPUT STRING
4754
4755 .ENABL LSB
4756
4757 015006 TESTND: ;CHECK FOR A DELIMITER IN INPUT STRING
4758 ;R4-->ASCII STRING
4759 ;RETURN WITH C BIT CLEAR IF NEXT CHARACTER
4760 ;IN THE STRING IS ONE OF THE FOLLOWING:
4761 ; 1)A SPACE,TAB,SLASH,COMMA,EQUAL SIGN,!,*,OR 'a'
4762 ; 2)A '+',OR'-', FOLLOWED BY AN ELEMENT OF '1'
4763 015006 010046 MOV R0,-(SP)
4764 015010 012700 015070' MOV #TESTLS,R0 ;POINT R0 TO LIST FOR '1' ABOVE
4765 015014 105710 10$: TSTB (R0) ;TEST FOR END OF LIST
4766 015016 001403 BEQ 20$ ;BR IF LIST 1 TRAVERSED WITH NO MATCH
4767 015020 121420 CMPB (R4),(R0)+ ;CHECK INPUT STRING AGAINST LIST 1
4768 015022 001374 BNE 10$ ;BR IF NOT A MATCH
4769 015024 000415 BR 50$ ;CLEAR C BIT AND EXIT
4770
4771 015026 012700 015103' 20$: MOV #SPCLST,R0 ;POINT R0 TO LIST FOR '2' ABOVE
4772 015032 105710 30$: TSTB (R0) ;CHECK FOR END OF LIST
4773 015034 001412 BEQ 60$ ;BR IF LIST '2' TRAVERSED WITH NO MATCH
4774 015036 121420 CMPB (R4),(R0)+ ;CHECK AN ITEM OF LIST '2' AGAINST INPUT STRING
4775 015040 001374 BNE 30$ ;BR IF NO MATCH
4776 015042 012700 015070' MOV #TESTLS,R0 ;POINT R0 TO LIST '1' AGAIN
4777 015046 105710 40$: TSTB (R0) ;CHECK FOR END OF LIST
4778 015050 001404 BEQ 60$ ;BR IF AT END OF LIST '1'
4779 015052 126420 000001 CMPB 1(R4),(R0)+ ;CHECK FOR A MATCH FROM LIST '1'
4780 015056 001373 BNE 40$ ;BR IF NO MATCH
4781 015060 005727 50$: TST (PC)+ ;CLEAR C BIT
4782 015062 000261 60$: SEC ;SET C BIT TO INDICATE NO MATCH
4783 015064 012600 MOV (SP)+,R0
4784 015066 000207 RTS PC
4785
4786 015070 040 011 015 TESTLS: .BYTE 40,11,15,54,57,72,100,41,52,'=',0
4787 015073 054 057 072
4788 015076 100 041 052
4789 015101 075 000
015103 053 055 000 SPCLST: .BYTE '+','- ',0
.EVEN
.DSABL LSB

```


4791
4792
4793 015106
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820

.SBTTL RECOGNIZE AND CONVERT A NUMERIC ASCII STRING

RECNUM: ;RECOGNIZE A <NUMBER> AND CONVERT TO BINARY
;A <NUMBER> IS A STRING OF ASCII DIGITS TERMINATED BY <BLANK>
;COMMA,<EOL>,OR <COMMENT>.
;THE DIGITS OF A <NUMBER> ARE 0 THRU 7 IF DEFAULT RADIX IS OCTAL,
;OR THE ABOVE PLUS 8,9,A,B,C,D,E,F IF THE DEFAULT RADIX IS HEX.
;A <NUMBER> MAY ALSO INCLUDE A LOCAL RADIX OVERRIDE.
;A LOCAL-RADIX-OVERRIDE IS A 'x' FOLLOWED BY AN 'O' OR AN 'X'.
;A LOCAL OVERRIDE MUST NOT BE SEPARTED FROM THE DIGITS OF A
;<NUMBER> BY ANY SPACES OR TABS.

;INPUTS:
; R0,R1,R2 ARE SCRATCH
; R3-->CURRENT NODE OF SYNTAX CHECK TREE
; R4-->INPUT STRING
; R5-->ROOT OF SYNTAX CHECK TREE
; 'DEFRAD' IS CURRENT DEFAULT RADIX, 0 IF HEX, 1 IF OCTAL

;OUTPUTS:
; 1) IF INPUT STRING IS RECOGNIZED AS A NUMBER:
; TRANSLATED NUMBER IS STORED AS DIRECTED BY 'INFO(R3)'
; C BIT IS CLEAR
; R4-->NEXT CHARACTER OF INPUT STRING PAST THE <NUMBER>
; 2) IF INPUT STRING NOT RECOGNIZED AS A <NUMBER>
; R4 IS UNCHANGED
; DATA AREA POINTED TO BY 'INFO(R3)' IS MUNGED
; C BIT IS SET

; R3,R5 NOT MODIFIED BY THIS ROUTINE

				.ENABL	LSB	
4822						
4823						
4824	015106	010446			MOV R4,-(SP)	;SAVE R4 IN CASE A <NUMBER> NOT RECOGNIZED
4825	015110	011300			MOV (R3),R0	;POINT R0 TO OUTPUT AREA
4826	015112	016301	000002		MOV ACTION(R3),R1	;R1 GETS # OF WORDS IN OUTPUT AREA
4827	015116	005020		10\$:	CLR (R0)+	;CLEAR OUTPUT AREA
4828	015120	005301			DEC R1	
4829	015122	003375			BGT 10\$	
4830	015124	116767	020273	005356	MOVB DEFRAD,TMPRAD	;PUT DEFAULT RADIX CODE IN TEMPORARY
4831	015132	004767	177650		JSR PC,TESTND	;TEST FOR A DELIMITER IN INPUT STRING
4832	015136	103317			BCC NOMATC	;BR IF FIRST CHARACTER IS A DELIMITER
4833	015140	121427	000055		CMPB (R4),#'-'	;CHECK FOR A LEADING MINUS SIGN
4834	015144	001004			BNE 50\$;BR IF NOT A MINUS SIGN
4835	015146	105724			TSTB (R4)+	;INCREMENT POINTER PAST MINUS SIGN
4836	015150	052767	000200	020222	BIS #NEGATE,TCONTL	;REMEMBER STRING IS TO BE NEGATED
4837	015156	121427	000045	50\$:	CMPB (R4),#'x	;CHECK FOR RADIX OVERRIDE
4838	015162	001014			BNE 30\$;BR IF NOT
4839	015164	105067	005320		CLRB TMPRAD	;ASSUME RADIX WILL BE HEX
4840	015170	005204			INC R4	
4841	015172	121427	000130		CMPB (R4),#'X	;CHECK FOR HEX OVERRIDE
4842	015176	001405			BEQ 25\$;BR IF IT WAS HEX
4843	015200	121427	000117		CMPB (R4),#'0	;SEE IF OVERRIDE IS OCTAL
4844	015204	001274			BNE NOMATC	;BR IF NOT OCTAL
4845	015206	105267	005276		INCB TMPRAD	;SET RADIX TO OCTAL
4846	015212	005204		25\$:	INC R4	;BUMP STRING POINTER PAST OVERRIDE
4847	015214	004767	177566	30\$:	JSR PC,TESTND	;CHECK FOR A DELIMITER IN INPUT STRING
4848	015220	103053			BCC 60\$;BR IF A DELIMITER SEEN
4849	015222	111401			MOVB (R4),R1	;PUT CHARACTER IN R1
4850	015224	012702	000003		MOV #3,R2	;ASSUME RADIX IS OCTAL(NEED 3 SHIFTS)
4851	015230	105767	005254		TSTB TMPRAD	;CHECK FOR OCTAL
4852	015234	001020			BNE 32\$;BR IF IS OCTAL
4853	015236	005202			INC R2	;ADD 1 TO SHIFT COUNT
4854	015240	121427	000106		CMPB (R4),#'F	;CHECK FOR A THRU F SINCE RADIX IS HEX
4855	015244	003254			BGT NOMATC	;BR IF GREATER THAN AN F(CAN NOT BE A DIGIT)
4856	015246	121427	000101		CMPB (R4),#'A	
4857	015252	002403			BLT 31\$;BR IF LESS THAN AN A
4858	015254	062701	000011		ADD #11,R1	;PRELIMINARY CONVERSION FOR A THRU F
4859	015260	000414			BR 33\$	
4860						
4861	015262	121427	000070	31\$:	CMPB (R4),#'8	;TEST FOR DIGITS 8 OR 9
4862	015266	001411			BEQ 33\$;BR IF AN 8
4863	015270	121427	000071		CMPB (R4),#'9	;CHECK FOR 9
4864	015274	001406			BEQ 33\$;BR IF 9
4865	015276	121427	000060	32\$:	CMPB (R4),#'0	;CHECK FOR DIGITS 0 THRU 7
4866	015302	002635			BLT NOMATC	;BR IF LESS THAN ASCII 0(CAN NOT BE A DIGIT)
4867	015304	121427	000067		CMPB (R4),#'7	
4868	015310	003232			BGT NOMATC	;BR IF > ASCII 7(CAN NOT BE A DIGIT ALLOWED)
4869	015312	042701	177760	33\$:	BIC #177760,R1	;CLEAR EXTRANEIOUS BITS
4870	015316	016346	000002	35\$:	MOV ACTION(R3),-(SP)	;PUT # OF WORDS IN OUTPUT AREA ON STACK
4871	015322	011300			MOV (R3),R0	;POINT R0 TO OUTPUT AREA
4872	015324	000241			CLC	
4873	015326	006120		40\$:	ROL (R0)+	;SHIFT THE OUTPUT AREA R2 TIMES
4874	015330	005316			DEC (SP)	;CHECK FOR ALL WORDS SHIFTED
4875	015332	003375			BGT 40\$;BR IF ALL WORDS NOT SHIFTED
4876	015334	005726			TST (SP)+	;POP COUNT FROM STACK

4877	015336	005302		DEC	R2		;REDUCE SHIFT COUNT
4878	015340	003366		BGT	35\$;BR IF MORE SHIFTS NEEDED
4879	015342	011300		MOV	(R3),R0		;R0 GETS POINTER TO OUTPUT AREA AGAIN
4880	015344	060110		ADD	R1,(R0)		;ADD IN INPUT'D DIGIT
4881	015346	000721		BR	25\$		
4882							
4883	015350	032767	000200 020022 60\$:	BIT	#NEGATE,TCONTL		;TEST FOR NEGATION OF NUMBER
4884	015356	001611		BEQ	MATCH		;EXIT IF NO NEGATION SPECIFIED
4885	015360	042767	000200 020012	BIC	#NEGATE,TCONTL		;CLEAR NEGATION FLAG
4886	015366	011300		MOV	(R3),R0		;R0 GETS POINTER TO CONVERTED STRING
4887	015370	016301	000092	MOV	ACTION(R3),R1		;R1 GETS # OF WORDS IN NUMBER
4888	015374	010102		MOV	R1,R2		;SAVE # OF WORDS FOR SUCCEEDING STEP
4889	015376	005120	70\$:	COM	(R0)+		;FIRST DO ONE'S COMPLEMENT OF NUMBER
4890	015400	005301		DEC	R1		
4891	015402	003375		BGT	70\$		
4892	015404	011300		MOV	(R3),R0		;GET POINTER TO R0 AGAIN
4893	015406	072720	000001	ADD	#1,(R0)+		;INCREMENT THE NUMBER BY ONE
4894	015412	065302	80\$:	DEC	R2		
4895	015414	003002		BGT	90\$;BR IF MORE CARRIES TO ADD
4896	015416	000167	177360	JMP	MATCH		
4897	015422	005520	90\$:	ADC	(R0)+		;ADD CARRY
4898	015424	000772		BR	80\$		
4899							
4900				.DSABL	LSB		
4901							
4902							;END OF PARSING MODULE

```

4904          .SBTTL  MAIN SYNTAX CHECK TREE
4905
4906          015426'      XXT=      ;SET THIS TO ROOT ADDRESS FOR TREE GENERATOR
4907
4908 015426      MAJTREE:      ;INFO  ACTION  YES      NO      LIST/NUMBER/ROUTINE CALL
4909
4910 015426      MTREPE: SEN    COCNTP, NULL, MTEOL, MTREPE ;PX-03-05
4911 015434      MTREPE: SEN    COREPE, SETRPT, MTSTAR, M1STAR
4912
4913 015442      MTSTAR: SEN    COSTAR, DOSTAR, MTNUM0, MTWAIT, ACT
4914 015450      MTNUM0: SEN    EFFADR, 4, MTEOL, MTSAL2, NUM
4915 015456      MTSAL2: SEN    SYBLST, SYBACT, MTEOL, 0, LST
4916
4917 015464      MTWAIT: SEN    COWAIT, NULL, MTDONE, MTNEXT
4918 015472      MTDONE: SEN    NCDONE, DOWAIT, MTEOL, 0, ACT
4919
4920 015500      MTNEXT: SEN    CONEXT, DONEXT, MTNUM1, MTINIT, ACT
4921 015506      MTNUM1: SEN    COUNT, 2, MTEOL, MTEOL, NUM
4922
4923 015514      MTINIT: SEN    COINIT, DOINIT, MTEOL, MTBOOT, ACT
4924
4925 015522      MTBOOT: SEN    COBOOT, SVBOOT, MTEOL, MTSHOW
4926
4927 015530      MTSHOW: SEN    COSHOW, DOSHOW, MTVERS, MTHALT, ACT
4928
4929 015536      MTVERS: SEN    NCVERS, DOSHVR, MTEOL, MTEOL, ACT
4930
4931 015544      MTHALT: SEN    COHALT, DOHALT, MTEOL, MTEXAM, ACT
4932
4933 015552      MTEXAM: SEN    COEXAM, SVEXAM, MTSALO, MTCONT
4934 015560      MTSALO: SEN    SYBLST, SYBACT, MTEOL, MTNUM2, LST
4935 015566      MTNUM2: SEN    EFFADR, 2, MTEOL, MTEXIR, NUM
4936 015574      MTEXIR: SEN    NCIR, DOIR, MTEOL, MTFIXA, ACT
4937 015602      MTFIXA: SEN    RESADD, NULL, MTEOL, MTEOL, RTN ;THIS NODE CALLS 'RESADD'
4938
4939 015610      MTCONT: SEN    COCONT, DOCONT, MTEOL, MTDEPO, ACT
4940
4941 015616      MTDEPO: SEN    CODEPO, SVDEPO, MTNUM3, MTQCLE
4942 015624      MTNUM3: SEN    EFFADR, 2, MTEQU, MTSAL1, NUM
4943 015632      MTSAL1: SEN    SYBLST, SYBACT, MTEQU, 0, LST
4944 015640      MTEQU: SEN    NCEQU, NULL, MTNUM4, MTNUM4
4945 015646      MTNUM4: SEN    DATATO, 4, MTEOL, 0, NUM
4946
4947 015654      MTQCLE: SEN    COQCLE, DOQCLE, MTNUM2, MTSET, ACT
4948
4949 015662      MTSET: SEN    COSET, NULL, MTSTEP, MTTEST
4950 015670      MTSTEP: SEN    NCSTEP, DOSSTN, MTSTOP, MTRELO, ACT
4951 015676      MTSTOP: SEN    STOPLS, STACLS, MTEOL, MTEOL, LST
4952 015704      MTRELO: SEN    NCRELO, NULL, MTCOL2, MTTERM
4953 015712      MTCOL2: SEN    NCCOLO, NULL, MTNUM5, MTEOL
4954 015720      MTNUM5: SEN    RELOCA, 2, MTEOL, 0, NUM
4955 015726      MTTERM: SEN    NCTERM, NULL, MTFILL, MTDEFA
4956 015734      MTFILL: SEN    NCFILL, DOSTER, MTCOLO, MTPROG, ACT
4957 015742      MTCOLO: SEN    NCCOLO, NULL, MTNUM4, 0
4958 015750      MTDEFA: SEN    NCDEFA, DOSTDF, MTDFOP, MTSOMO, ACT

```

4959 015756	MTPROG: SEN	NCPROG, DOSTPG, MTEOL, 0, ACT
4960 015764	MTDFOP: SEN	DFUPLS, DFOPAC, MTCOM0, MTEOL, LST
4961 015772	MTCOMG: SEN	NCCOMM, NULL, MTDFOP, MTEOL
4962 016000	MTSOH0: SEN	NCSOMM, DOSTSO, MTEOL, MTCLOC, ACT
4963 016006	MTCL0C: SEN	NCCLOC, DOSTCN, MTCLOP, 0, ACT
4964 016014	MTCL0P: SEN	CLOPLS, CLOPAC, MTEOL, MTEOL, LST

			;INFO	ACTION	YES	NO	LIST/NUMBER/ROUTINE	CALL
4966								
4967								
4968	016022	MTTEST: SEN	COTEST.	DOTEST.	MTEOL.	MTWCS.	ACT	
4969								
4970	016030	MTWCS: SEN	COWCS.	DOWCS.	MTEOL.	MTREBO.	ACT	
4971								
4972	016036	MTREBO: SEN	COREBO.	DOREBO.	MTEOL.	MTUNJA.	ACT	
4973								
4974	016044	MTUNJA: SEN	COUNJA.	DOUNJA.	MTEOL.	MTLOAD.	ACT	
4975								
4976	016052	MTLOAD: SEN	COLOAD.	SVLOAD.	MTDX1.	MTCLEA		
4977								
4978	016060	MTCLEA: SEN	COCLEA.	NULL.	MTCLKP.	MTINDI		
4979	016066	MTCLKP: SEN	CLEOPL.	CLEOPA.	MTEOL.	0.	LST	
4980								
4981	016074	MTINDI: SEN	COINDI.	DOINDI.	MTDX1.	MTHelp.	ACT	
4982								
4983	016102	MTHelp: SEN	COHELP.	SVHELP.	MTEOL.	MTPERF		
4984								
4985	016110	MTPERF: SEN	COPERF.	DOPERF.	MTEOL.	MTLINK.	ACT	
4986								
4987	016116	MTLINK: SEN	COLINK.	DOLINK.	MTEOL.	MTOVER.	ACT	
4988								
4989	016124	MTOVER: SEN	COOVER.	DOOVER.	MTDX1.	MTENAB.	ACT	
4990								
4991	016132	MTEOL: SEN	EOLLST.	EOLACT.	0.	0.	LST	
4992								
4993	016140	MTDX1: SEN	NCDX1.	NULL.	MTCOL1.	MTXLAT		
4994	016146	MTCOL1: SEN	NCCOLO.	SETDX1.	MTXLAT.	0		
4995								
4996	016154	MTXLAT: SEN	XLATFN.	NULL.	MTEOL.	0.	RTN	;THIS NODE CALLS 'XLATFN'
4997								
4998	016162	MTENDX: SEN	NCDX1.	DOENDX.	MTCOL3.	0.	ACT	
4999	016170	MTCOL3: SEN	NCCOLO.	NULL.	MTEOL.	MTEOL		
5000								
5001								
5002								
5003	016176	MTENAB: SEN	COENAB.	NULL.	MTTALK.	MTDISA		
5004	016204	MTTALK: SEN	NCTALK.	ENTTLK.	MTEOL.	MTLOCA.	ACT	
5005	016212	MTLOCA: SEN	NCLOCA.	NULL.	MTCNTL.	MTREMO		
5006	016220	MTCNTL: SEN	NCCNTL.	ENLOCN.	MTEOL.	MTCOPY.	ACT	
5007	016226	MTCOPY: SEN	NCCOPY.	ENLOCO.	MTEOL.	MTFLP1.	ACT	
5008	016234	MTFLP1: SEN	NCFLOP.	ENFLOP.	MTEOL.	0.	ACT	
5009	016242	MTECHO: SEN	NCECHO.	ENECHO.	MTEOL.	MTCARR.	ACT	
5010	016250	MTCARR: SEN	NCCARR.	NULL.	MTERR0.	MTENDX		
5011	016256	MTERR0: SEN	NCERRO.	ENCLER.	MTEOL.	0.	ACT	
5012	016264	MTREMO: SEN	NCREMO.	NULL.	MTFPR1.	MTECHO		
5013	016272	MTFPR1: SEN	NCFLOP.	ENREMT.	MTEOL.	0.	ACT	
5014								
5015	016300	MTDISA: SEN	CODISA.	NULL.	MTECH1.	MTXLOA		
5016	016306	MTECH1: SEN	NCECHO.	DSECHO.	MTEOL.	MTCAR1.	ACT	
5017	016314	MTCAR1: SEN	NCCARR.	NULL.	MTERR1.	MTLOC1		
5018	016322	MTERR1: SEN	NCERRO.	DSCLER.	MTEOL.	0.	ACT	
5019	016330	MTLOC1: SEN	NCLOCA.	NULL.	MTCP01.	MTREM1		
5020	016336	MTCP01: SEN	NCCOPY.	DSLOCO.	MTEOL.	MTFLP2.	ACT	

5021 016344	MTFLP2: SEN	NCFLOP, DSFLOP, MTEOL, 0,	ACT
5022 016352	MTREM1: SEN	NCREMO, NULL, MTFPR2, 0	
5023 016360	MTFPR2: SEN	NCFLOP, DSREMT, MTEOL, 0,	ACT
5024			
5025 016366	MTXLOA: SEN	COXLGA, DOXLOA, MTNUM6, MTEOL,	ACT
5026 016374	MTNUM6: SEN	EFFADR, 4,	NUM
5027 016402	MTNUM7: SEN	COUNT, 4,	NUM
5028			
5029	000212	APTCMD=.-MTENAB	
5030			

```
5032 .SBTTL QUALIFIER SYNTAX CHECK TREE
5033
5034 016410' XXT=. ;SET THIS TO ROOT ADDRESS OF TREE FOR TREE GENERATOR
5035
5036 016410 QALTRE: ;INFO ACTION YES NO LIST/NUMBER
5037
5038 016410 SEN CONEXT, SETNEX, QTCOLO, QTCOMM
5039 016416 QTCOLO: SEN NCCOLO, NULL, QTNUM0, QTTSND
5040 016424 QTNUM0: SEN NEXTCT, 1, 0, QTTSND, NUM
5041
5042 016432 QTCOMM: SEN NCCOMD, SETCOM, 0, QTWCS
5043
5044 016440 QTWCS: SEN NCWCS, SETWCS, 0, QTSTAR
5045
5046 016446 QTSTAR: SEN COSTAR, NULL, QTCOL3, QTDFOP
5047 016454 QTCOL3: SEN NCCOLO, NULL, QTNUM1, QTTSND
5048 016462 QTNUM1: SEN EFFADR, 2, 0, QTSALO, NUM
5049 016470 QTSALO: SEN SYBLST, SYBACT, 0, 0, LST
5050
5051 016476 QTDFOP: SEN DFOPLS, DFOPAC, QTCOM2, 0, LST
5052 016504 QTCOM2: SEN NCCOMM, NULL, QTDFOP, QTTSND
5053
5054 016512 QTTSND: SEN TESTND, NULL, 0, 0, RTN ;THIS NODE CALLS 'TESTND'
```


22-ESKAA-10.1 MAINTREE AND QUALIFIER TREE LISTS
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 66
 MAINTREE AND QUALIFIER TREE LISTS

```

5056 .SBTTL MAINTREE AND QUALIFIER TREE LISTS
5057
5058 ;RECOGNITION LISTS AND ASSOCIATED ACTION LISTS FOR BOTH
5059 ;THE MAIN SYNTAX TREE AND THE QUALIFIER SYNTAX TREE
5060
5061 ;A RECOGNITION LIST IS A SERIES OF WORD POINTERS TO RECOGNITION
5062 ;STRINGS, TERMINATED BY A 0 WORD.
5063 ;THE ACTIONS ASSOCIATED WITH EACH RECOGNITION STRING IN A LIST
5064 ;FOLLOW IN ORDER IMMEDIATELY AFTER THE END OF THE RECOGNITION LIST.
5065
5066 ;END-OF-LINE LIST
5067 016520 020361' 020364' 000000 EOLLST: .WORD NCEOL, NCCMNT, 0
5068 016526 002764' 002764' EOLACT: .WORD EXECUT,EXECUT
5069
5070 ;SET STEP OPTION LIST
5071 016532 020142' 020320' 020054' STOPLS: .WORD NCINST,NCSTAT,NCBUS,NCNORM,0
5072 016540 020163' 000000
5072 016544 000000C 000000C 000000C STACLS: .WORD DOSSTI!1,DOSSTS!1,DOSSTB!1,DOSSTN!1
5073 016552 000000C
5074
5075 ;'SET CLOCK' OPTION LIST
5075 016554 020163' 020112' 020306' CLOPLS: .WORD NCNORM,NCFAST,NCSLOW,0
5076 016562 000000
5076 016564 000000C 000000C 000000C CLOPAC: .WORD DOSTCN!1,DOSTCF!1,DOSICS!1
5077
5078 ;'CLEAR' OPTION LIST
5079 016572 020312' 020324' 000000 CLEOPL: .WORD NCSOMM,NCSTEP,0
5080 016600 000000C 000000C CLEOPA: .WORD DOCLSO!1,DOSSTN!1
5081
5082 ;SYMBOLIC ADDRESS LIST
5083 016604 020362' 020363' 020176' SYBLST: .WORD NCASTK,NCPLUS,NCPSL,NCMNUS,COINDI,NCPC,NCSP
5084 016612 020365' 017756' 020173'
5084 016620 020316'
5084 016622 020214' 020217' 020222' .WORD NCR0,NCR1,NCR2,NCR3,NCR4,NCR5,NCR6,NCR7,NCR8,NCR9
5084 016630 020225' 020230' 020233'
5084 016636 020236' 020241' 020244'
5084 016644 020247'
5085 016646 020252' 020256' 020262' .WORD NCR10,NCR11,NCR12,NCR13,NCR14,NCR15,NCAP,NCFP,0
5085 016654 020266' 020272' 020276'
5085 016662 020051' 020124' 000000
5086 016670 017560' 017574' 017426' SYBACT: .WORD SETLSA,SETPLS,SETPSL,SETMNS,SETLSD,SETPC,SETSP
5086 016676 017576' 017704' 017434'
5086 016704 017442'
5087 016706 017544' 017542' 017540' .WORD SETR0,SETR1,SETR2,SETR3,SETR4,SETR5,SETR6,SETR7,SETR8,SETR9
5087 016714 017536' 017534' 017532'
5087 016722 017530' 017526' 017524'
5087 016730 017522'
5088 016732 017520' 017516' 017514' .WORD SETR10,SETR11,SETR12,SETR13,SETR14,SETR15,SETR12,SETR13
5088 016740 017512' 017510' 017506'
5088 016746 017514' 017512'
5089
5090 ;'SET DEFAULT' OPTION LIST(ALSO USED BY QUALIFIERS)
5091 016752 020201' 020344' 020127' DFOPLS: .WORD NCPHYS,NCVIRT,NCGENE,NCINTE,NCIDBU,NCCONS
5091 016760 020146' 020136' 020073'
5092 016766 020334' 020133' 020167' .WORD NCVBUS,NCHEX,NCCTA,NCBYTE,NCWORD,NCLONG,NCQUAD,0

```

	016774	020057'	020353'	020157'	
	017002	020211'	000000		
5093	017006	017324'	017322'	017320'	DFOPAC: .WORD SETPHY,SETVIR,SETGEN,SETINT,SETIDB,SETCON
	017014	017316'	017314'	017312'	
5094	017022	017310'	017402'	017400'	.WORD SETVBU,SETHex,SETOCT,SETBYT,SETWRD,SETLNG,SETQAD
	017030	017372'	017370'	017366'	
	017036	017364'			

5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107

.SBTTL PARSE ACTION ROUTINES

;THESE ROUTINES ARE CALLED BY 'RECOG' ROUTINE
;VIA A 'JSR PC,(R2)'

;R3-->CURRENT NODE OF 'MAINTREE' OR 'QALTRE'
;R4-->NEXT LEXEME OF INPUT STRING
;R5-->ROOT NODE OF MAINTREE OR QALTRE
;R0 IS SCRATCH
;R1 IS CLEAR(USED BY SET DEFAULT ACTIONS)
;R2-->WHATTODO (USED BY ALL 'SVXXXX' ROUTINES)

```

5109 .SBTTL ACTIONS THAT SAVE OPERATION TO PERFORM
5110
5111 .ENABL LSB
5112
5113 017040 012712 003104' SVBOOT: MOV #DOBOOT,(R2) ;SAVE 'BOOT'
5114 017044 004767 175554 JSR PC,REMLEA ;SLUFF SPACES AND TABS FROM INPUT STRING
5115 017050 004767 175732 JSR PC,TESTND ;TEST FOR A DELIMITER IN INPUT STRING
5116 017054 103027 BCC STBOFL ;BR IF A DELIMITER SEEN
5117 017056 012702 022472' MOV #BOOSTR,R2 ;POINT R2 TO ASCII BOOT FILE NAME STRING
5118 017062 112762 000060 000002 MOVB #'0,2(R2) ;ASSUME UNIT ZERO
5119 017070 000403 BR 50$
5120
5121 017072 004767 175710 40$: JSR PC,TESTND ;TEST FOR A DELIMITER IN THE INPUT STRING
5122 017076 103005 BCC 60$ ;BR IF DELIMITER IN INPUT STRING
5123 017100 112422 50$: MOVB (R4)+,(R2)+ ;PUT CHARACTER INTO BOOT NAME STRING
5124 017102 005201 INC R1 ;KEEP TRACK OF HOW MANY CHARACTERS
5125 017104 020127 000003 CMP R1,#3 ;SEE IF 3 CHARACTERS YET
5126 017110 002770 BLT 40$ ;BR TO GET ANOTHER IF NOT 3 YET
5127 017112 010446 60$: MOV R4,-(SP) ;SAVE INPUT STRING POINTER
5128 017114 012704 022472' MOV #BOOSTR,R4 ;POINT R4 TO BOOT FILENAME STRING
5129 017120 004767 172652 JSR PC,XLATFN ;TRANSLATE TO RAD50
5130 017124 012604 MOV (SP)+,R4 ;REPLACE INPUT STRING POINTER
5131 017126 103011 BCC 90$ ;BR IF XLATION OK
5132 017130 105267 016260 INCB ABORT ;THIS WILL PREVENT AN ATTEMPT TO BOOT
5133 017134 012700 017166' STBOFL: MOV #DEFNAM,R0 ;SET UP DEFAULT BOOT FILENAME
5134 017140 012701 022546' SETFIL: MOV #FILENM,R1
5135 017144 012021 MOV (R0)+,(R1)+
5136 017146 012021 MOV (R0)+,(R1)+
5137 017150 012021 MOV (R0)+,(R1)+
5138 017152 000207 90$: RTS PC
5139
5140 017154 012712 003140' SVHELP: MOV #DOINDI,(R2) ;HELP FILE IS INDIRECT FILE
5141 017160 012700 017174' MOV #HELNAM,R0 ;POINT R0 TO HELP FILE NAME BLOCK
5142 017164 000765 BR SETFIL ;PUT HELP FILE NAME IN 'FILENM'
5143
5144 017166 DEFNAM: ;DEFAULT BOOT FILE NAME IN RAD50
5145 017166 014716 .RAD50 \DEF\
5146 017170 007347 .RAD50 \BOO\
5147 017172 012314 .RAD50 \CMD\
5148
5149 017174 HELNAM: ;CONSOLE HELP FILE NAME IN RAD50
5150 017174 012446 .RAD50 \CON\
5151 017176 074444 .RAD50 \SOL\
5152 017200 031760 .RAD50 \HLP\
5153
5154 017202 ECONAM: ;WCS ECO FILE NAME IN RAD50
5155 017202 110113 .RAD50 \WCS\
5156 017204 134745 .WORD 134745 ;NOTE: THIS IS A 'WILD CARD' TO THE FILE OPEN RTN
5157 017206 062074 .RAD50 \PAT\
5158
5159 017210 RESNAM: ;AUTO-RESTART INDIRECT COMMAND FILE NAME IN RAD50
5160 017210 070533 .RAD50 \RES\
5161 017212 076472 .RAD50 \TAR\
5162 017214 012314 .RAD50 \CMD\
5163

```

```

5164 017216 005727          SVEXAM: TST      (PC)+          ;CLEAR C BIT FOR EXAMINE
5165 017220 000261          SVDEPO: SEC              ;SET C BIT FOR DEPOSIT
5166 017222 005567 016164    ADC      DEEXBY          ;C BIT GOES TO DEEXBY TO REMEMBER EX OR DE
5167 017226 012712 004514'   MOV      #DODEEX,(R2)      ;REMEMBER EX/DE ROUTINE
5168 017232 016767 017346 003266 MOV      EFFADR,SAVEFF ;SAVE UPDATED EFFECTIVE ADDRESS
5169 017240 016767 017342 003262 MOV      EFFADR+2,SAVEFF+2
5170 017246 000207          RTS      PC
5171
5172 017250 012712 012322'   SVLOAD: MOV      #DOLOAD,(R2)    ;SAVE 'LOAD'
5173 017254 105267 016132    INCB     DEEXBY          ;FORCE DEPOSIT
5174 017260 005067 017320    CLR      EFFADR          ;CLEAR LOAD START ADDRESS
5175 017264 005067 017316    CLR      EFFADR+2
5176 017270 000207          RTS      PC
5177
5178          .DSABL  LSB
5179
5180 017272 105267 016117    SETRPT: INCB     RPTFLG          ;SET REPEAT FLAG
5181 017276 000207          RTS      PC
5182
5183 017300 052767 000040 016072 SETDX1: BIS      #DX1FLG,TCONTL ;SET DRIVE 1 FLAG
5184 017306 000207          RTS      PC
5185
5186          .SBTTL  ACTIONS FOR QUALIFIERS AND SET DEFAULT COMMAND
5187
5188
5189          ;THE FOLLOWING ROUTINES REQUIRE R1=0 ON ENTRY
5190
5191
5192          ;ROUTINE TO SET CURRENT ADDRESS SPACE BYTE
5193
5194
5201 017310 005201          SETVBU: INC      R1
5202 017312 005201          SETCON: INC      R1
5203 017314 005201          SETIDB: INC      R1
5204 017316 005201          SETINT: INC      R1
5205 017320 005201          SETGEN: INC      R1
5206 017322 005201          SETVIR: INC      R1
5207 017324 110167 016072    SETPHY: MOVB     R1,CURADS
5208 017330 000207          RTS      PC
5209 017332          RESADD: ;RESTORE CONTENTS OF 'EFFADR' FROM 'SAVEFF'
5210 017332 016767 003170 017244 MOV      SAVEFF,EFFADR
5211 017340 016767 003164 017240 MOV      SAVEFF+2,EFFADR+2
5212 017346 105767 016050    TSTB     CURADS          ;ADDRESS SPACE SPECIFIED?
5213 017352 100003          BPL      10$              ;BR IF YES
5214 017354 116767 003126 016040 MOVB     LASADS,CURADS ;USE PREVIOUS ADDRESS SPACE
5215 017362 000207          10$:  RTS      PC

```

```

5217 ;ROUTINE TO SET CURRENT DATA LENGTH BYTE
5218
5219 .ENABL LSB
5220
5221 ;REQUIRES R1=0 ON ENTRY
5222
5227
5228 017364 005201 SETQAD: INC R1
5229 017366 005201 SETLNG: INC R1
5230 017370 005201 SETWRD: INC R1
5231 017372 110167 016023 SETBYT: MOVB R1,CURLNH
5232 017376 000207 RTS PC
5233
5234 ;SET RADIX BYTE
5236
5237 017400 005201 SETOCT: INC R1
5238 017402 110167 016012 SETHEX: MOVB R1,CURRAD
5239 017406 000207 RTS PC
5240
5241
5242 017410 005267 015770 SETNEX: INC NEXTCT ;SET /NEXT COUNT TO DEFAULT OF 1
5243 017414 000207 RTS PC
5244
5245 017416 052767 100000 015756 SETCOM: BIS #COMQAL,MICFLG ;SET COMMAND MODE BIT FOR MICRO-DIAGNOSTICS
5246 017424 000207 RTS PC
5247
5248 017426 004067 000016 SETPSL: JSR R0,SETUPR ;SET UP TO ACCESS THE PROCESSOR STATUS LONG WORD
5249 017432 004 017 .BYTE IDBSPC,17
5250
5251 017434 004067 000010 SETPC: JSR R0,SETUPR ;SET UP TO ACCESS THE PC
5252 017440 002 017 .BYTE GENSPC,17
5253
5254 017442 004067 000002 SETSP: JSR R0,SETUPR ;SET UP TO ACCESS THE STACK POINTER
5255 017446 002 016 .BYTE GENSPC,16
5256
5257 017450 112067 015746 SETUPR: MOVB (R0)+,CURADS ;SET ADDRESS SPACE CODE
5258 017454 112000 MOVB (R0)+,R0 ;R0 GETS ADDRESS
5259 017456 112767 000002 015735 MOVB #LNGLNH,CURLNH ;SET LENGTH TO LONG WORD
5260 017464 005726 TST (SP)+ ;POP SAVED R0 OFF STACK
5261 017466 000501 BR SETOUT ;FINISH UP BELOW
5262
5263 017470 052767 100000 015702 SETWCS: BIS #WCSDES,TCONTL ;REMEMBER PRESENCE OF /WCS QUALIFIER
5264 017476 052767 010000 017100 BIS #10000,EFFADR ;FORCE LOAD ADDRESS TO START OF WCS
5265 017504 000207 RTS PC
5266
5267 .DSABL LSB

```

```
5269 .SBTTL SYMBOLIC REGISTER ADDRESS SETUPS
5270
5271 .ENABL LSB
5272
5273 017506 005201 SETR15: INC R1
5274 017510 005201 SETR14: INC R1
5275 017512 005201 SETR13: INC R1
5276 017514 005201 SETR12: INC R1
5277 017516 005201 SETR11: INC R1
5278 017520 005201 SETR10: INC R1
5279 017522 005201 SETR9: INC R1
5280 017524 005201 SETR8: INC R1
5281 017526 005201 SETR7: INC R1
5282 017530 005201 SETR6: INC R1
5283 017532 005201 SETR5: INC R1
5284 017534 005201 SETR4: INC R1
5285 017536 005201 SETR3: INC R1
5286 017540 005201 SETR2: INC R1
5287 017542 005201 SETR1: INC R1
5288 017544 110167 005007 SETR0: MOVB R1,10$
5289 017550 005001 CLR R1
5290 017552 004067 177672 JSR R0,SETUPR
5291 017556 002 .BYTE GENSPC
5292 017557 000 10$: .BYTE 0
5293
5294 .DSABL LSB
```

;SAVE ADDRESS

;SET UP GEN REG ADD SPACE

;NOTE:MUST BE READ/WRITE*****

```

5296 .SBTTL ACTIONS FOR SYMBOLIC ADDRESSES
5297
5298 .ENABL LSB
5299
5300 017560 116767 002722 015634 SETLSA: MCVB LASADS,CURADS
5301 017566 012700 036572' MOV #LASADD,R0 ;POINT R0 TO 'LAST ADDRESS' USED
5302 017572 000446 BR 50$
5303
5304 017574 005727 SETPLS: TST (PC)+ ;SET ADDRESS TO 'NEXT' ADDRESS
5305 017576 000261 SETMNS: SEC ;SET ADDRESS TO 'PRECEEDING' ADDRESS
5306 017600 016700 016766 MOV LASADD,R0 ;R0 GETS LAST ADDRESS USED
5307 017604 016701 016764 MOV LASADD+2,R1 ;R1 GETS MSB'S
5308 017610 004767 165522 JSR PC,SETLNH ;SET 'LNHDATA' TO LENGTH OF DATA IN BYTES
5309 017614 116702 015602 MOVVB CURADS,R2 ;R2 GETS CURRENT ADDRESS SPACE
5310 017620 100004 BPL 20$ ;BR IF ADDRESS SPACE SPECIFIED BY QUALIFIER
5311 017622 116702 002660 MOVVB LASADS,R2 ;USE LAST SPACE INSTEAD
5312 017626 110267 015570 MOVVB R2,CURADS ;SET CURRENT ADDRESS TO LAST ADDRESS SPACE
5313 017632 103405 20$: BCS 30$ ;BR IF TO DECREMENT ADDRESS
5314 017634 006302 ASL R2
5315 017636 067200 005266' ADI @ADUPTB(R2),R0 ;UPDATE ADDRESS CORRECT AMOUNT
5316 017642 005501 AD R1 ;R1 GETS ANY CARRY
5317 017644 000407 BR 40$
5318
5319 017646 006302 30$: ASL R2
5320 017650 167200 005266' SUB @ADUPTB(R2),R0 ;REDUCE ADDRESS BY PROPER AMOUNT
5321 017654 005601 SBC R1 ;REDUCE R1 BY CARRY
5322 017656 052767 000400 015514 BIS #MINSAD,TCONTL ;REMEMBER THE BACKWARD ADDRESSING
5323 017664 105767 015525 40$: TSTB RPTFLG ;TEST FOR REPEAT SET
5324 017670 001333 BNE SETLSA ;BR IF REPEAT, SET ADDRESS BACK
5325 017672 010067 016706 SETOUT: MOV R0,EFFADR ;SET UP CURRENT ADDRESS
5326 017676 010167 016704 MOV R1,EFFADR+2
5327 017702 000207 RTS PC
5328
5329 017704 012700 022466' SETLSD: MOV #LASDAT,R0 ;USE LAST DATA AS NEXT ADDRESS
5330 017710 012701 036604' 50$: MOV #EFFADR,R1 ;POINT R1 TO ADDRESS
5331 017714 012021 MOV (R0)+,(R1)+ ;SET UP ADDRESS
5332 017716 012021 MOV (R0)+,(R1)+
5333 017720 000207 RTS PC
5334
5335 .DSABL LSB

```


5337
5338
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363 017722
5364 017726
5365 017732
5366 017736
5367 017743
5368 017747
5369 017753
5370 017756
5371 017757
5372 017763
5373 017767
5374 017773
5375 017777
5376 020004
5377 020010
5378 020014
5379 020020
5380 020024
5381 020027
5382 020032
5383 020036
5384 020041
5385 020044
5386 020047
5387
5388
5389
5390 020051
5391 020054
5392 020057
5393 020063
5394 020067
5395 020073
5396 020077
5397 020103
5398 020106
5399 020112
5400 020115
5401 020120

100

.SBTTL REGOGNITION STRINGS

;NOTE: EACH CHECK STRING MUST BE OF THE FORM 'RST X,Y' WHERE BOTH X AND Y
; ARE NON-BLANK.
; 'X' IS THE CHARACTERS THAT MUST MATCH THE INPUT STRING
; 'Y' IS THE CHARACTERS THAT MAY FOLLOW 'X' IN THE INPUT STRING,BUT
; ARE NOT REQUIRED FOR RECOGNITION. IF THE INPUT STRING
; DOES CONTAIN THESE CHARACTER THEN THEY MUST MATCH.
;EXCEPTION: CHECK STRING THAT BEGIN WITH A DELIMITER(SEE 'TESTND' FOR
; THE DEFINITION OF A DELIMITER). NEED BE ONLY ONE CHARACTER LONG.

;COMMAND NAME STRINGS

COBOOT: RST B,OOT
COCLEA: RST CL,EA
COCONT: RST C,ONT
CODEPO: RST D,EPOS
COEXAM: RST E,XAM
COHALT: RST H,ALT
COHELP: RST HE,L
COINDI: .BYTE 'a
COINIT: RST I,NIT
COLINK: RST LI,NK
COLOAD: RST L,OAD
CONEXT: RST N,EXT
COOVER: RST O,VERL
COPERF: RST P,ERF
COQCLE: RST Q,CLE
COREBO: RST REB,O
COREPE: RST R,EPE
COSET: RST SE,T
COSHOW: RST SH,O
COSTAR: RST S,TAR
COTEST: RST T,ES
COUNJA: RST U,NJ
COWAIT: RST WA,I
COWCS: RST W,C

;NON-COMMAND RECOGNITION STRINGS

NCAP: RST AP,X
NCBUS: RST B,US
NCBYTE: RST B,YTE
NCCLOC: RST C,LOC
NCCOMD: RST C,OMM
NCCONS: RST CO,NS
NCDEFA: RST D,EFA
NCDONE: RST D,ON
NCDX1: RST DX1,:
NCFAST: RST F,AS
NCFILL: RST F,IL
NCFLOP: RST F,LOP

5402	020124	NCFP:	RST	FP,X
5403	020127	NCGENE:	RST	G,ENE
5404	020133	NCHEX:	RST	H,EX
5405	020136	NCIDBU:	RST	ID,BU
5406	020142	NCINST:	RST	I,NST
5407	020146	NCINTE:	RST	I,NTERN
5408	020154	NCIR:	RST	IR,X
5409	020157	NCLONG:	RST	L,ONG
5410	020163	NCNORM:	RST	N,ORM
5411	020167	NCOCTA:	RST	O,CTA
5412	020173	NCPC:	RST	PC,R
5413	020176	NCPSL:	RST	P,SL
5414	020201	NCPHYS:	RST	P,HYS
5415	020205	NCPRQG:	RST	P,ROG
5416	020211	NCQUAD:	RST	Q,UA
5417	020214	NCR0:	RST	R0,X
5418	020217	NCR1:	RST	R1,X
5419	020222	NCR2:	RST	R2,X
5420	020225	NCR3:	RST	R3,X
5421	020230	NCR4:	RST	R4,X
5422	020233	NCR5:	RST	R5,X
5423	020236	NCR6:	RST	R6,X
5424	020241	NCR7:	RST	R7,X
5425	020244	NCR8:	RST	R8,X
5426	020247	NCR9:	RST	R9,X
5427	020252	NCR10:	RST	R10,X
5428	020256	NCR11:	RST	R11,X
5429	020262	NCR12:	RST	R12,X
5430	020266	NCR13:	RST	R13,X
5431	020272	NCR14:	RST	R14,X
5432	020276	NCR15:	RST	R15,X
5433	020302	NCRELO:	RST	R,ELO
5434	020306	NCSLOW:	RST	S,LOW
5435	020312	NCSOMM:	RST	SO,MM
5436	020316	NCSP:	RST	S,P
5437	020320	NCSTAT:	RST	S,TAT
5438	020324	NCSTEP:	RST	S,TEP
5439	020330	NCTERM:	RST	T,ERM
5440	020334	NCVBUS:	RST	VB,US
5441	020340	NCVERS:	RST	V,ERS
5442	020344	NCVIRT:	RST	V,IRT
5443	020350	NCWCS:	RST	WC,S
5444	020353	NWORD:	RST	W,ORD

;NOTE:THE R IS A DUMMY USED TO SATISFY

;MISCELLANEOUS AND PUNCTUATION STRINGS

5445			
5446			
5447			
5448	020357	054	NCCOMM: .BYTE 54
5449	020360	072	NCCOLO: .BYTE :
5450	020361	015	NCEOL: .BYTE 15
5451	020362		NCASTK: RST *
5452	020363		NCPLUS: RST +
5453	020364		NCCMNT: RST !
5454	020365		NCMNUS: RST -
5455	020366	075	NCEQU: .BYTE '=
5456	020367	020 200	COCNTP: .BYTE 20,200 ; (CONTROL-P)

ZZ-ESKAA-10.1 REGOGNITION STRINGS
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 72-2
REGOGNITION STRINGS

5457

5458

5459

5460 020371

5461 020377

5462 020406

5463 020413

5464 020416

5465 020423

5466 020432

5467 020436

5468 020442

5469 020451

5470 020456

5471

5472

000073

;APT SPECIFIC CHECK STRINGS

COENAB: RST EN,ABLE
CODISA: RST DI,SABLE
COXLOA: RST X,LOAD
NCTALK: RST T,AL
NCLOCA: RST L,OCAL
NCCNTL: RST CON,TROL
NCCOPY: RST COP,Y
NCECHO: RST E,CHO
NCCARR: RST C,ARRIER
NCERRO: RST E,RROR
NCREMO: RST R,EMOTE
APTSTR=.-COENAB
.EVEN

```

5474
5499
5500 020464          CONVER: SPMES  \PVER,\SVER,\PEDT,\SED
5501 020475          BOTING: MES    <      (BOOTING)>
5502 020510          AUTRES: MES    <      (AUTO-RESTART)>
5503 020530          WCSLOD: MES    <      (RELOADING WCS)>
5504 020551          ALRDHA: MES    <      CPU HALTED>
5505 020565          HLTMES: MES    <      HALTED AT >
5506 020601          STRRUN: MES    <      ?CPU NOT IN CONSOLE WAIT LOOP>
5507 020640          TMEOUT: MES    <      ?NO CPU RESPONSE>
5508 020662          CANTDO: MES    <      ,FUNCTION ABORTED>
5509 020794          OPNPAR: MES    <      ( )
5510 020710          CLSPAR: MES    <      >>
5511 020712          DASH:  MES    <      ->
5512 020714          TWOSPC: MES    <      >
5513
5514 020717          UNKERR: MES    <      ?MIC-ERR, CODE=>
5515 020737          MEMMAN: MES    <      ?MEM-MAN FAULT, CODE=>
5516 020765          CONSER: MES    <      ?MIC-ERR ON FUNCTION>
5517 021013          RESCOM: MES    <      INIT SEQ DONE>
5518 021032          INSTIV: MES    <      ?INT-STK INVLD>
5519 021052          CPDBLE: MES    <      ?CPU DBLE-ERR HLT>
5520 021075          ILIEVC: MES    <      ?ILL I/E VEC>
5521 021113          NOWCSU: MES    <      ?NO USR WCS>
5522 021130          EINTPE: MES    <      INT PENDING>
5523 021145          HLINST: MES    <      HALT INST EXECUTED>
5524 021171          ERRCHM: MES    <      ?CHM ERR>
5525 021203          ERRPRG: MES    <      ?INT-REG ERR>
5526 021221          MMTMOU: MES    <      ?MICRO-MACHINE TIME OUT>
5527
5528 021252          TAB:  MES    <      >
5529 021254          CPUIS: MES    <      CPU >
5530 021262          RUNNIN: MES    <      RUNNING>
5531 021272          HLTED: MES    <      HALTED>
5532 021301          SOMMIS: MES    <      ,SOMM >
5533 021310          ISSET: MES    <      SET>
5534 021314          ISCLR: MES    <      CLEAR>
5535 021322          STPEQU: MES    <      ,STEP=>
5536 021331          STINST: MES    <      INST>
5537 021336          STBUS:  MES    <      BUS>
5538 021342          STSTA: MES    <      STAT>
5539 021347          NRMALL: MES    <      NONE>
5540 021354          CLKEQU: MES    <      ,CLOCK=>
5541 021364          CLKNOR: MES    <      NORM>
5542 021371          CLKFAS: MES    <      FAST>
5543 021376          CLKSLO: MES    <      SLOW>
5544 021403          ORADIX: MES    <      OCT>
5545 021407          OHEX:  MES    <      HEX>
5546 021413          SPHY:  MES    <      PHYS>
5547 021420          SVIR:  MES    <      VIRT>
5548 021425          SGEN:  MES    <      GEN>
5549 021431          SINT:  MES    <      INT>
5550 021435          SIDB:  MES    <      IDBU>
5551 021442          SCON:  MES    <      CONS>
5552 021447          SVBU:  MES    <      VBUS>

```

5553 021454	ADDEQU: MES	< ,ADD=>	
5554 021462	RADEQU: MES	< RAD=>	
5555 021470	DATEQU: MES	< ,DAT=>	
5556 021476	FILLEQU: MES	< ,FILL=>	
5557 021505	RELEQU: MES	< ,REL=>	
5558 021513	DBYT: MES	< BYTE>	
5559 021520	DWRD: MES	< WORD>	
5560 021525	DLNG: MES	< LONG>	
5561 021532	DQAD: MES	< QUAD>	
5562			
5563 021537	PHYIDN: MES	< P >	
5564 021544	GENIDN: MES	< G >	
5565 021551	INTIDN: MES	< I >	
5566 021556	IDBIDN: MES	< ID >	
5567 021563	CONIDN: MES	< C >	
5568 021570	VBUIDN: MES	< VB >	
5569 021575	IRIDN: MES	< IR >	
5570 021602	PSLSTR: MES	< >	
5571			
5572 021605	CLKERR: MES	< ?>	
5573 021610	CLOCKS: MES	< CPU CLK STOP>	
5574 021625	UPCEQU: MES	< UPC=>	
5575 021633	APCEQU: MES	< APC=>	
5576 021641	CPTN: MES	< CPT0>	;NOTE THIS MESSAGE MUST BE IN R/W STORAGE(NEVER ROM)
5577			
5578 021647	CRMESQ: MES	< ?'>	
5579 021653	ISANER: MES	< ' IS INCORRECT>	
5580 021672	ISINCO: MES	< ' IS INCOMPLETE>	
5581			
5582 021712	FLNMER: MES	< ?FILE NAME ERR>	
5583 021732	NOSUFL: MES	< ?FILE NOT FOUND>	
5584 021753	FLPERR: MES	< ?FLOPPY ERR, CODE=>	
5585 021776	LOISDN: MES	< LOAD DONE, >	
5586 022013	BYTESL: MES	< BYTES LOADED>	
5587 022031	MICWSL: MES	< MICROWORDS LOADED>	
5588			
5589 022054	DISERR: MES	< ?CANT DISABLE BOTH FLOPPIES>	
5590 022110	NOTREM: MES	< ?REMOTE ACCESS NOT SUPPORTED>	
5591 022146	XERR1: MES	< ?COMMAND>	
5592 022157	XERR2: MES	< ?DATA>	
5593 022165	XERR3: MES	< CHKSUM ERR>	
5594 022201	WCNEFP: MES	< ?WARNING-WCS & FPLA VER MISMATCH>	
5595 022242	WCNEPC: MES	< ?FATAL-WCS & PCS VER MISMATCH>	
5596 022300	PCSEQU: MES	< VER: PCS=>	
5597 022313	WCSEQU: MES	< WCS=>	
5598 022321	FPLEQU: MES	< FPLA=>	
5599 022330	GHMES: MES	< KE780 PRESENT>	; for pre-release comment out
5600	;SPEC1: MES	< WARNING:: PRE-RELEASE CONSOLE >	; include this
5601 022347	CONEQU: MES	< CON=>	
5602			
5603 022355	BADLIN: MES	< ?IND-COM ERR>	
5604 022373	INDEX1: .BYTE	9., '<', 'a', 'E', 'X', 'I', 'T', '>', 15, 12	
022376	101	074	100
022401	105	130	111
022404	124	076	015
012			

ZZ-ESKAA-10.1 TEXT STRING STORAGE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 73-2
TEXT STRING STORAGE

5605	022405	006	074	100	EOFMES: .BYTE	6,'<','a','E','O','F','>
	022410	105	117	106		
	022413	076				
5606						
5607	022414	005	015	012	CONPMP: .BYTE	5,15,12,'>','>','>
	022417	076	076	076		
5608	022422	005	015	012	LNKPMP: .BYTE	5,15,12,'<','<','<
	022425	074	074	074		
5609						
5610	022430	002			CRMES: .BYTE	2
5611	022431	015	012		TIMTRP: .BYTE	15,12
5612	022433	077	124	122	.ASCII	\?TRAP-4,RESTARTING CONSOLE\
	022436	101	120	055		
	022441	064	054	122		
	022444	105	123	124		
	022447	101	122	124		
	022452	111	116	107		
	022455	040	103	117		
	022460	116	123	117		
	022463	114	105			
5613					TIMEND=.	
5614	022465				.EVEN	

ZZ-ESKAA-10.1 TEMPORARY STORAGE
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 74
 TEMPORARY STORAGE

```

5616                      .SBTTL  TEMPORARY STORAGE
5617
5618          000000          HEXRAD=0          ;WARNING: DO NOT CHANGE THESE DEFINITIONS WITHOUT CAREFUL THOUGHT
5619          000000          PHYSPC=0
5620          000001          VIRSPC=1
5621          000002          GENSPC=2
5622          000003          INTSPC=3
5623          000004          IDBSPC=4
5624          000005          CONSPC=5
5625          000006          VBUSPC=6
5626
5627          000000          BYTLNH=0
5628          000001          WRDLNH=1
5629          000002          LNGLNH=2
5630          000003          QADLNH=3
5631
5632 022466 000000 000000          LASDAT: .WORD 0,0          ;LAST DATA USED BY EXAM OR DEPO
5633 022472          000          000          060 BOOSTR: .BYTE 0,0,'0','B','O','O','.',',','C','M','D',15
      022475          102          117          117
      022500          056          103          115
      022503          104          015
5634
5635 022505          000          CNVTDN: .BYTE 0
5636 022506          000          LASADS: .BYTE 0          ;ADDRESS SPACE CODE FOR CONTENTS OF 'EFFADR'
5637 022507          000          ERRCCD: .BYTE 0          ;USED FOR PRINTING ERROR CODES
5638 022510          000          TMPRAD: .BYTE 0
5639 022511          000          NOECHO: .BYTE 0          ;BOOT ECHO SUPPRESSION FLAG
5640 022512          000          LINKNG: .BYTE 0          ;NON-ZERO WHEN LINKING IN PROGRESS
5641 022513          000          SAWTMO: .BYTE 0          ;NON-ZERO WHEN MICROMACHINE TIME OUT SEEN AND REPORTED
5642 022514          000          LODFLG: .BYTE 0          ;'LOAD-A-FILE' FLAG (EDIT-21B)
5643                      .EVEN
5644 022516 000000          LNHDAT: .WORD 0          ;CURRENT DATA LENGTH IN BYTES
5645 022520 000000          LNHCOD: .WORD 0          ;CODED DATALENGTH FOR MICRO-ROUTINES
5646 022522 000000 000000          RELOCA: .WORD 0,0          ;RELOCATION REGISTER
5647 022526 000000 000000          SAVEFF: .WORD 0,0
5648 022532 000000 000000          SAVCOD: .WORD 0,0          ;SAVES 'HALT REASON' CODE FOR AUTO-RESTARTS(V02-01.
5649
5650          ;
5651          ; THIS TABLE WAS DELETED IN VERSION 6.1. A VBUS EXAMINE WILL NOW PRINT 16(D)
5652          ; BYTES OF DATA INDEPENDENT OF THE CHANNEL NUMBER
5653          ;
5654          ;VBUSCD:          ;TABLE OF VBUS CHANNEL LENGTHS IN BYTES
5655          ;          .BYTE 15          ;CH 0
5656          ;          .BYTE 10          ;CH 1
5657          ;          .BYTE 15          ;CH 2
5658          ;          .BYTE 14          ;CH 3
5659          ;          .BYTE 16          ;CH 4
5660          ;          .BYTE 20          ;CH 5
5661          ;          .BYTE 6          ;CH 6
5662          ;          .BYTE 10          ;CH 7
5663
5664          ;LOADER TEMPS
5665 022536 000000          CURRSEC: .WORD 0          ;CURRENT FLOPPY SECTOR
5666 022540 000000          SECSLF: .WORD 0          ;# OF SECS REMAINING IN FILE
5667 022542 000000          BUFFRP: .WORD 0          ;POINTER INTO SECTOR BUFFER

```

5668	022544	000000		BYTSLF: .WORD	0	;# BYTES LEFT IN SECTOR BUF
5669	022546			BYTSLD:		
5670	022546	000000	000000	FILENM: .WORD	0,0	;FILE NAME STORAGE(RAD50)
5671	022552	000000		EXTENS: .WORD	0	;EXTENSION
5672	022554	000000		FILPNT: .WORD	0	
5673						
5674						
5675	022556	000000				
5676	022560	000000		INDBYT: .WORD	0	;PNTR INTO INDIRECT COMMAND BUFFER
5677	022562	000000		INDLFT: .WORD	3	;# SECS LEFT IN IND FILE
5678				INDSEC: .WORD	0	;CURRENT SECTOR IN BUFFER
5679						;.....TO HERE
5680	022564	000000		SECLD: .WORD	0	;REMEMBERS SECTOR IN IND BUF
5681						


```

5683                ;CONSOLE DEFINITIONS FOR TEMPORARY STORAGE
5684
5685                022600                USRBUF=22600        ;256 BYTE BUFFER (22600 TO 23177)
5686                000400                USRBSZ=256.         ;SIZE OF USER BUFFER IN BYTES
5687
5688
5689
5690 022566          FILLTO 23200
5691
5692                ;NOTE: BUF0 IS PLACED ON A 128 BYTE BOUNDARY FOR OVERLAY
5693                ;      CUT-OFF PURPOSES(SEE 'CUTOFF','LASTOR')
5694
5695 023200          BUF0: ;INDIRECT COMMAND FILE BUFFER
5696                ;NOTE: CONSOLE INTERRUPT VECTOR CONTENTS ARE STORED HERE FOR
5697                ;      USE IMMEDIATELY AFTER A CONSOLE BOOTSTRAP OR OVERLAY
5698 023200 140056    .WORD 0THRTP                ;0
5699 023202 000340    .WORD 340                  ;2
5700 023204 013710'  .WORD ODDADD                ;4
5701 023206 000000    .WORD 0                   ;6
5702 023210 140056    .WORD 0THRTP                ;10
5703 023212 000340    .WORD 340                  ;12
5704
5705                .REPT 2
5706
5707
5708 023224 000026    .WORD .-BUF0+2              ;24(POWER FAIL)
5709 023226 000000    .WORD 0                   ;26
5710 023230 140016    .WORD EMTSER              ;30
5711 023232 000340    .WORD 340                  ;32
5712
5713
5714                .REPT 5
5715
5716
5717 023260 032356'  .WORD KBDINT                ;60
5718 023262 000340    .WORD 340                  ;62
5719 023264 032300'  .WORD PRTINT              ;64
5720 023266 000340    .WORD 340                  ;66
5721
5722
5723                .REPT 2
5724
5725
5726 023300 140004    .WORD CLKSER                ;100
5727 023302 000340    .WORD 340                  ;102
5728
5729
5730                .REPT 15.
5731
5732
5733 023400          BUF1: ;FLOPPY BUFFER SHARED BY CONSOLE PROGRAM AND FILE SERVICES
5734                .REPT 13.
5735
5736 023464 140026    .WORD DXPREI                ;264(FLOPPY)
5737 023466 000340    .WORD 340                  ;266
5738
5739                .REPT 2
5740
5741
5742 023500 032560'  .WORD CTXINT                ;300(CIB-TX READY)
5743 023502 000340    .WORD 340                  ;302
5744 023504 140012    .WORD CRXINT              ;304(CIB-RX DONE)
5745 023506 000340    .WORD 340                  ;306
5746

```

5783	023510	036414'	.WORD	RTIINS
5784	023512	000340	.WORD	340
5785	023514	036414'	.WORD	RTIINS
5786	023516	000340	.WORD	340
5788				
5789		000014	.REPT	12.
5793		000113	LASTOR=<.-BASE-1000>/200	
5794				

;COMPUTES MAX # OF SECTORS LOADABLE BY OVERLAY

ZZ-ESKAA-10.1 TEMPORARY STORAGE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 80
TEMPORARY STORAGE

B 11

20-MAY-1986

Fiche 1 Frame B11

Sequence 131

7985 023600

FILLTO 32000

```
8354
8355 032000 000207          ENTTLK: RTS      PC
8356 032002          TSTHLF: ;PART OF 'KLUDG2'*****!!!!!!!!!!!!!!!!!!!!*****
8357 032002 020127 036324'    CMP      R1,#LTEH5f    ;ARE WE LOADING LOCAL ECHO BUFFER?
8358 032006 001006          BNE      10$             ;BR IF NOT
8359 032010 005767 004116    TST      WRTQUE         ;ANY MESSAGES BEING PRINTED?
8360 032014 001003          BNE      10$             ;BR IF YES
8361 032016 112767 000001 004264 MOVB    #1,ECHOIN    ;ENTER 'ECHO IN PROGRESS' STATE
8362 032024 000207          10$:    RTS      PC
```

```
8367 .SBTTL
8368 .SBTTL  CONSOLE SWITCH POSITION CHECKER
8369
8370 ;+
8371 ;THIS ROUTINE CHECKS FOR A CHANGE IN THE POSITION OF THE CONSOLE MODE SWITCH
8372 ;THE ALGORITHM USED IS AS FOLLOWS
8373 ;
8374 ; IF<LAST POSITION OF SWITCH=CURRENT POSITION> THEN<EXIT> ELSE<ENTER NEW MODE>
8375 ;
8376 ;CONSOLE MODE CHANGE ACTIONS:
8377 ;
8378 ;CHANGING THE POSITION OF THE CONSOLE MODE SWITCH WILL
8379 ;HAVE THE FOLLOWING EFFECTS:
8380 ;
8381 ; A) CONSOLE SWITCH ENTERS 'LOCAL' POSITION:
8382 ; 1)DISABLE 'LOCAL COPY','LOCAL CONTROL','CARRIER ERROR REPORTING',
8383 ; AND 'TALK MODE ECHO'.
8384 ;
8385 ; B) CONSOLE SWITCH ENTERS 'LOCAL-DISABLE' POSITION:
8386 ; 1) SAME AS 'A-1' ABOVE
8387 ; 2) FORCE PROGRAM I/O MODE
8388 ; 3) CLEAR REMOTE TERMINAL INTERRUPT ENABLES
8389 ; 4) CLEAR 'DATA TERMINAL READY' ON REMOTE INTERFACE
8390 ;
8391 ; C) CONSOLE SWITCH ENTERS 'REMOTE-DISABLE' POSITION
8392 ; 1) FORCE PROGRAM I/O MODE
8393 ; 2) ENABLE INTERRUPTS FROM REMOTE TERMINAL INTERFACE
8394 ; 3) ASSERT 'DATA TERMINAL READY' ON REMOTE INTERFACE
8395 ; 4) FORCE 'LOCAL COPY' MODE FOR CUSTOMER SECURITY
8396 ;
8397 ; D) CONSOLE SWITCH ENTERS 'REMOTE' POSITION
8398 ; 1) ENABLE INTERRUPTS FROM REMOTE TERMINAL INTERFACE
8399 ; 2) ASSERT 'DATA TERMINAL READY' ON REMOTE INTERFACE
8400 ;
8401 ;-
8402 ;NOTE: MODE BITS IN 'MCS' REGISTER (BITS 0 AND 1)
8403 ; 0=LOCAL,1=LOCAL-DISABLE,2=REMOTE,3=REMOTE DISABLE
8404 ;
8405 ;NOTE: CONTENTS OF R0 ARE DESTROYED BY THIS ROUTINE
8406
8407 032026 CHKSWH: ;CONSOLE MODE SWITCH TRANSITION CHECKER
8408 032026 010146 MOV R1,-(SP)
8409 032030 013700 173034 MOV @#MCS,R0 ;R0 GETS MCS REGISTER CONTENTS.
8410 032034 042700 177774 BIC #177774,R0 ;CLEAR ALL EXCEPT MODE BITS.
8411 032040 105767 004720 TSTB SETSWH ;FORCE A SET-UP REGARDLESS ?
8412 032044 001014 BNE 10$ ;YES, GO DO IT.
8413 032046 120067 037750 CMPB R0,LASPOS ;LAST POSITION=CURRENT?
8414 032052 001420 BEQ 30$ ;BR IF NO CHANGE DETECTED.
8415 032054 005001 CLR R1 ;INIT 500 MS TIMER.
8416 032056 005201 5$: INC R1 ;STALL FOR 500 MS TO ENSURE THAT WE ARE NOT
8417 032060 000240 NOP ;STILL MOVING KEYSWITCH. OTHERWISE WE MAY DO
8418 032062 000240 NOP ;A FALSE SET UP FOR A DISABLE POSITION.
8419 032064 001374 BNE 5$
8420 032066 013700 173034 MOV @#MCS,R0 ;GET CURRENT SWITCH MODE AGAIN.
8421 032072 042700 177774 BIC #177774,R0 ;SAVE ONLY THE MODE BITS.
```

```
8422 032076 105067 004662      10$: CLRB   SETSWH      ;RESET FORCE SET-UP FLAG.
8423 032102 110067 037750'      MOVB   RO,LASPOS    ;SAVE NEW MODE AS LAST MODE
8424 032106 006300              ASL     RO              ;USE NEW MODE AS OFFSET
8425 032110 004770 032142'      JSR     PC,aMODCHG(R0) ;INTO SETUP ROUTINE
8426 032114 105067 037751'      30$: CLRB   AUTFLG      ;INIT AUTO RESTART FLAG
8427 032120 032737 000004 173034 BIT     #AUTORS,a#MCS ;TEST AUTO-RESTART SWITCH BIT
8428 032126 001403              BEQ     40$             ;BR IF AUTO-RESTART OFF
8429 032130 112767 000377 037751' MOVB   #377,AUTFLG    ;MAKE SOFT AUTO RESTART FLAG REFLECT SWITCH
8430 032136 012601              40$: MOV     (SP)+,R1
8431 032140 000207              LOCOUT: RTS   PC
8432
8433 032142 032160' 032152' 032270' MODCHG: .WORD ELOCAL,ELOCDS,EREMOT,EREMDS
      032150 032250'
```

3435 .SBTTL CONSOLE SWITCH MODE CHANGE

;CONSOLE MODE SWITCH TRANSITION SET-UP ROUTINES

ELOCDS: .ENABL LSB
;ENTER LOCAL DISABLE MODE
INCB PGM10M ;ENABLE PROGRAM I/O MODE
BR 20\$;SKIP THE LOCAL ENTRY DELAY

ELOCAL: ;NOTE: THE TIME DELAY INTRODUCED HERE BEFORE PERFORMING THE
; SET-UP FOR ENTRY INTO 'LOCAL' MODE IS TO ALLOW FOR
; THE FACT THAT THE CONSOLE MODE SWITCH READS AS 'LOCAL'
; WHEN THE SWITCH IS BETWEEN VALID POSITIONS. THIS
; DELAY OF ONE-HALF SECOND GIVES A PERSON TIME TO SWITCH
; BETWEEN 2 VALID POSITIONS, WITHOUT GETTING A SET-UP
; FOR 'LOCAL' MODE IN BETWEEN POSITIONS.(V0108 EDIT).
; IF, DURING THE DELAY, THE CONSOLE DETECTS THAT THE MODE
; HAS CHANGED FROM 'LOCAL' TO SOME OTHER MODE, THE SET-UP
; IS ABORTED.
;TIMING: THE LOOP BELOW STARTING AT '10\$', UP TO BUT NOT INCLUDING
; '20\$', TAKES 19.95 MICRO-SECONDS TO EXECUTE (+ OR - 20%).
; THEREFORE R0 IS INITIALLY SET UP TO -25062 (10) TO PROVIDE
; A DELAY OF (19.95 * 25062) = 500 MILLI-SECONDS. EDIT (03-14-80)

SWCTIM=-25062.

8468	032214	105267	003437	10\$:	MOV	#SWCTIM,R0	.R0 GETS DELAY CONSTANT FOR 1/2 SECOND
8469	032220	001775	037772	20\$:	BIT	#REMOTLOCKD,R0	;HAS MODE CHANGED FROM LOCAL?
8470	032222	005077	037766	30\$:	BNE	LOCCUT	;BR IF YES AND ABORT SET-UP
8471	032226	042767	101400		INC	R0	;TIME DELAY DONE?
8472	032226	052767	002000		BNE	10\$;BR IF NOT
8473	032226	105067	004024		TSTB	NOREM	;REMOTE INTERFACE THERE?
8474	032234	000207			BEQ	ELOCKX	;BR IF IT IS NOT
8475	032246	000207			BIC	#XMTINT,ARMXCSR	;DISABLE REMOTE XMIT INTERRUPTS.
8476					TSTB	ARMXCSR	;DONE SENDING CURRENT CHAR?
8477					BEQ	30\$;WAIT FOR IT TO COMPLETE.
8478					CLR	ARMXCSR	;DISABLE REMOTE INTR AND DTR/RTS
8479							
8480	032246	000207			CLRB	PROTOD	;CLEAR PROTOCOL MODE
8481							
8482							
8483	032250	105267	003341		BIC	#REMECH!TLKMOD!PRNINH!ROFLAG,TCTFLG	;CLR REMOTE OPTION.
8484	032250	042767	100600		BIS	#DISCAR!LOCCOP!LOCCNT,TCTFLG	;ENABLE LOCAL OPTION BITS
8485	032254	000207			CLRB	PROTOD	
8486							
8487							
8488	032262	052767	000000		RTS	PC	
8489	032270				.DSABL	LSB	
8490	032270						
8491	032276	000207					
8495							

ZZ-ESKAA-10.1 CONSOLE SWITCH MODE CHANGE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 85-1
CONSOLE SWITCH MODE CHANGE

G 11

20-MAY-1986

Fiche 1 Frame G11

Sequence 136

8580

8582 032300

8583

8584

8585

8586

8587

8588

8589

8590

8591

8592

8593

8594

8595

8596

8597

8598

8599

8600

8601

8602

8603

8604

8605

8606

8607 032300 105767 004004

8608 032304 001413

8609 032306 005767 004020

8610 032312 001004

8611 032314 105067 003770

8612 032320 000137 144474

8613

8614 032324 010046

8615 032326 010146

8616

8617

8618 032330 000137 144512

8619

8620

8621 032334 005767 003572

8622 032340 001367

8623 032342 005767 003764

8624 032346 001764

8625 032350 105267 003734

8626 032354 000763

8627

KLUDG2: ;THIS ROUTINE FIXES A PROBLEM INVOLVED WITH THE SEQUENCING OF
 ;ECHOES AND MESSAGES TO THE LOCAL TERMINAL. THE NORMAL
 ;CODE FOR THIS PROCESS IS IN ROM SO THE FIX MUST BE MADE HERE
 ;UNTIL A CHANGE IN ROM CAN BE MADE.
 ;
 ;BASICALLY THERE ARE TWO TYPES OF OUTPUT TO THE LOCAL TERMINAL
 ; A) 'MESSAGES' A STRING OF 1 OR MORE BYTES TO BE
 ; PRINTED AS ONE CONTIGUOUS STRING.
 ; B) 'ECHOES' SINGLE CHARACTERS TO BE PRINTED.
 ;
 ;THESE TWO TYPES OF OUTPUT ARE HANDLED BY DIFFERENT STRUCTURES:
 ; MESSAGES ARE KEPT IN A QUEUE, AND ECHOES ARE KEPT IN
 ; A RING BUFFER(FOR REASONS OF SYNCHRONIZATION BETWEEN
 ; LOCAL AND REMOTE TERMINAL RUNNING AT DIFFERENT SPEEDS.)
 ;
 ;THE PROBLEM BEING ADDRESSED BY THIS ROUTINE IS AS FOLLOWS:
 ; MESSAGES ARE GIVEN PRIORITY OVER ECHOES IN THE ROM ROUTINE
 ; THAT RESPONDS TO 'TX READY' INTERRUPTS. THIS WILL CAUSE
 ; A MESSAGE STRING TO APPEAR IN THE MIDDLE OF ECHOES IF
 ; THE MESSAGE GETS ISSUED WHILE ECHOES ARE IN PROGRESS.
 ; THIS ROUTINE CAUSES ECHOES TO FINISH BEFORE PRINTING A
 ; MESSAGE AND VICE VERSA.
 ;
 ;ENTER BELOW ON EACH 'TX READY' INTERRUPT FROM LOCAL TTY INTERFACE.

TSTB ECHOIN ;ECHOES IN PROGRESS?
 BEQ 20\$;BR IF NOT
 TST LTHBF+6 ;ANY FURTHER ECHOES TO DO?
 BNE 10\$;BR IF YES
 CLRB ECHOIN ;CLEAR 'ECHOES IN PROGRESS' FLAG
 5\$: JMP a#144474 ;ENTER ROM ROUTINE TO SERVICE INTERRUPT
 ;* (*PRTBGN*)
 10\$: MOV R0,-(SP) ;THIS DUPLICATES ROM ROUTINE ENTRY CONDITIONS
 MOV R1,-(SP) ;DITTO
 ;*****
 ;THIS IS THE MAJOR KLUDGE!(ENTERING ROM AT FIXED ADDRESS)
 JMP a#144512
 ;*****
 20\$: TST WRTQUE ;MESSAGES TO PRINT?
 BNE 5\$;BR IF YES TO ENTER ROM
 TST LTHBF+6 ;ANY ECHOES TO PUMP OUT?
 BEQ 5\$;BR IF NOT
 INCB ECHOIN ;SET 'ECHOES IN PROGRESS' FLAG
 BR 10\$;GO HANDLE ECHOS

```

8629
8630 032356
8631
8632
8633
8634
8635
8636
8637
8638
8639
8640
8641
8642
8643
8644
8645
8646
8647
8648
8649
8650
8651
8652
8653 032356 010046
8654 032360 017700 037760'
8655 032364 042700 000200
8656 032370 032767 000000 003134
8657 032376 001425
8658 032400 120027 000020
8659 032404 001011
8660 032406 004737 150070
8661 032412 001060
8662 032414 004777 003734
8663 032420 052767 002000 003104
8664 032426 000452
8665
8666 032430 004777 003722 2$:
8667 032434 032767 001000 003070
8668 032442 001444
8669 032444 004777 140072'
8670 032450 000441
8671
8672 032452 004737 150070 5$:
8673 032456 001407
8674 032460 032767 000000 003044
8675 032466 001003
8676 032470 004777 003516
8677 032474 000427
8678
8679 032476
8680 032476 105767 003113
8681 032502 001402
8682 032504 000137 144374
8683 032510 120027 000017

KLUDG3: ;*
;THIS ROUTINE FIXES TWO PROBLEMS WITH THE ROM KEYBOARD INTERRUPT
;SERVICE ROUTINE 'KBDBGN':
; 1) RAPIDLY-REPEATED CONTROL-C'S WOULD CAUSE AN UNEXPECTED
; TRAP. THE CONSOLE NO LONGER RECOGNIZES CONROL-C, WITH
; NO USER REQUEST ACTIVE, AS A REBOOT.
; 2) DUE TO A DISPARITY IN INTERPRETATION OF THE ORIGINAL
; INTENT, THE CONSOLE WOULD DISABLE 'TALK' ON RECEIPT
; OF A CONTROL-P FROM EITHER TERMINAL, DESPITE THE
; KEYSWITCH POSITION. NOW WE WILL DISABLE ON A CONTROL-P
; FROM THE LOCAL TERMINAL ONLY IF THE KEYSWITCH IS NOT IN
; REMOTE.(REMOTE TERMINAL RESPONSE NOT AFFECTED BY CHANGE.)
;IN ORDER FOR THIS ROUTINE TO WORK, WE HAVE TO ENTER ROM AT CERTAIN
;FIXED ADDRESSES.
;--
.ENABL LSB
;KEYBOARD INTERRUPT SERVICE.
;INTERRUPTS OFF THRU-OUT THIS ROUTINE.

MOV R0,-(SP)
MOV @RBUF,R0 ;GET THE CHARACTER AND STATUS BITS
BIC #200,R0 ;CLEAR PARITY BIT IF ANY
BIT #TLKMOD,TCTFLG ;IN TALK MODE?
BEQ 5$ ;BR IF NOT
CMPB R0,#20 ;CONTROL-P? (EXIT TALK MODE)
BNE 2$ ;BR IF NO
JSR PC,@#150070 ;IN REMOTE MODE?(TSTREM)
BNE 25$ ;BR IF YES
JSR PC,@EXTKPT ;CLEAR TALK MODE
BIS #DISCAR,TCTFLG ; DISABLE CARRIER ERROR REPORTING
BR 25$ ;EXIT

2$: JSR PC,@WRTRMP ;ECHO TO REMOTE LINE
BIT #REMECH,TCTFLG ;TALK ECHO ENABLED?
BEQ 25$ ;BR TO EXIT IF NO
JSR PC,@WRTLCP ;ECHO BACK TO LOCAL TERMINAL
BR 25$

5$: JSR PC,@#150070 ;IN REMOTE MODE?(TSTREM)
BEQ RMTENT ;BR IF NO
BIT #LOCCNT,TCTFLG ;LOCAL CONTROL?
BNE RMTENT ;BR IF YES
JSR PC,@CHKLCI ;TEST FOR PROTOCOL INTERRUPT CHARACTER
BR 25$ ;EXIT

RMTENT:
TSTB PGMIOM ;TEST FOR PROGRAM I/O MODE
BEQ 6$ ;BR IF NO
JMP @#144374 ;IF YES, GO TO 'PROGIO' IN ROM
6$: CMPB R0,#17 ;TEST FOR CONTROL-0

```

ZZ-ESKAA-10.1 CONSOLE SWITCH MODE CHANGE
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 87-1
CONSOLE SWITCH MODE CHANGE

8684	032514	001002		BNE	7\$;BR IF NOT
8685	032516	000137	144356	JMP	a#144356	;GO BACK TO ROM TO ECHO
8686	032522	105767	003004	7\$: TSTB	TCTFLG	;USER REQUEST ACTIVE?
8687	032526	100012		BPL	25\$;EXIT IF NOT
8688	032530	105767	037747'	TSTB	APTLOD	; DID APT LOAD US?(EDIT 4-08)
8689	032534	001405		BEQ	8\$; BRANCH IF NO(EDIT 4-08)
8690	032536	122700	000020	CMPB	#20,R0	; CONTROL P?(EDIT 4-08)
8691	032542	001002		BNE	8\$; BRANCH IF NO(EDIT 4-08)
8692	032544	112700	000003	MOVB	#3,R0	; FORCE TO CONTROL C(EDIT 4-08)
8693	032550	000137	146062	8\$: JMP	a#146062	;GO TO ROM TO SERVICE ('KBDSEr')
8694						
8695	032554	012600		25\$: MOV	(SP)+,R0	
8696	032556	000002		RTI		;RETURN FROM INTERRUPT
8697						
8698				.DSABL	LSB	
8699						

```

8701 ;CIB 'TX READY' INTERRUPT SERVICE -- FIX FROM 'CTXBGN' IN ROM
8702 ;
8703 ;NEW SOFTWARE COMMUNICATION CODES:
8704 ; VMS SENDS 'F03' TO CLEAR THE WARM-START FLAG
8705 ; 'F04' TO CLEAR THE COLD-START FLAG
8706 ;
8707 ;
8708 000017 SOFCOM=17 ;SOFTWARE COMMUNICATION CODE
8709 000003 CONEXM=3 ;'EXAMINE CONSOLE MEMORY' CODE
8710
8711 .ENABL LSB
8712
8713 ; EU00169 **9**
8714 ; THIS TXRENT ROUTINE ALLOWS US TO JMP TO ROM AND ENTER THE SENDST AND PUTWRD
8715 ; ROUTINES WITH THE INTERRUPTS ENABLED. THE SAME SCENARIO CAN HAPPEN HERE AS
8716 ; IT DID IN THE EU00257 PROBLEM REPORT WHERE THE TX RDY BIT MAY BE ACCESSED
8717 ; MULTIPLE TIMES IN THE SAME INTERRUPT CYCLE AND QUEUE MAY BECOME UNMANAGABLE.
8718
8719 032560 106427 000340 TXRENT: MTPS #340 ;DISABLE INTERRUPTS **9**
8720 032564 010046 MOV R0,-(SP) ;SAVE SCRATCH REGISTERS
8721 032566 010146 MOV R1,-(SP) ;
8722 032570 105767 003021 TSTB PGM10M ;IN PROGRAM I/O MODE?
8723 032574 001473 BEQ 30$ ;EXIT IF NOT
8724
8725 032576 005000 ENTFOR: CLR R0
8726 032600 005001 CLR R1
8727 032602 153701 173024 BISB a#FMIDLO,R1 ;R1 GETS DATA FROM 'FMID' REGISTER
8728 032606 153700 173025 BISB a#FMIDLO+1,R0 ;R0 GETS SELECT CODE
8729 032612 001433 BEQ 10$ ;BR IF SELECT CODE 0 (P I/O OUTPUT)
8730 032614 020027 000003 CMP R0,#CONEXM ;'EXAMINE MEMORY' CODE?
8731 032620 001420 BEQ 35$ ;GO DO IT, IF SO
8732 032622 020027 000017 CMP R0,#SOFCOM ;SOFTWARE COMMUNICATION CODE?
8733 032626 001402 BEQ 45$ ;GO DO IT
8734 032630 000137 145070 25$: JMP a#145070 ;GO TO SERVICE AS USUAL IN ROM.
8735
8736 032634 022701 000003 45$: CMP #3,R1 ;CLEAR WARM-START FLAG
8737 032640 001002 BNE 15$ 15$
8738 032642 105067 037745' CLRB WRMSTR
8739 ;BR 25$ ;(SAVE A FEW BYTES)
8740 032646 022701 000004 15$: CMP #4,R1 ;CLEAR COLD-START FLAG
8741 032652 001366 BNE 25$ ;RETURN TO ROM AS USUAL
8742 032654 105067 037746' CLRB CLDSTR
8743 032660 000763 BR 25$ ;EXIT BACK TO ROM ROUTINE
8744
8745 ;'EXAMINE CONSOLE MEMORY' SERVICE
8746 032662 066701 003504 35$: ADD BASEAD,R1 ;CALCULATE ADDRESS OF BYTE TO EXAMINE
8747 032666 005000 CLR R0
8748 032670 151100 BISB (R1),R0 ;LOWER BYTE OF R0 GETS CONTENTS OF ADDRESS
8749 032672 052700 001400 BIS #CONEXM*400,R0 ;UPPER BYTE GETS EXAMINE CODE BACK
8750 032676 000137 145134 JMP a#145134 ;GO TO SERVICE AS USUAL IN ROM.
8751
8752 032702 110167 002661 10$: MOVB R1,STARCR ;R1 GETS CHAR
8753 032706 105067 003361 CLRB SYNC ;CLEAR TERMINAL SYNC FLAG
8754 032712 004737 150070 JSR PC,a#150070 ;IN REMOTE MODE?(TSTREM)
8755 032716 001406 BEQ 12$ ;SKIP SETTING SYNC FLAG IF NOT

```

```

8756 032720 032767 000000 002604      BIT      #LOCCOP,TCTFLG ;LOCAL COPY SET?
8757 032726 001402                      BEQ      12$      ;SKIP SETTING SYNC FLAG,IF NOT
8758 032730 105267 003337                      INCB     SYNC
8759 032734                      12$:  T$WRIT  #STARCR,#1,#CHRPNT ;WRITE ONE BYTE AND RETURN BELOW
8760 032754 103003                      BCC      30$      ;BR IF NO ERROR
8761 032756 005726                      TST      (SP)+    ;POP ERROR OFF STACK
8762 032760 004767 000022                      JSR      PC,TXSETR ;SET TRANSMITTER READY
8763 032764 012601                      30$:  MOV      (SP)+,R1 ;RESTORE SCRATCH REGISTERS
8764 032766 012600                      MOV      (SP)+,R0
8765 032770 000002                      RTI
8766
8767 032772 105367 003275      CHRPNT: DECB     SYNC      ;RETURN TO HERE AFTER CHAR HAS PRINTED
8768 032776 002015                      BGE      40$      ;SKIP SETTING TX-READY UNTIL ALL DONE
8769 033000 105767 003721                      TSTB     MESFLG
8770 033004 001012                      BNE      40$
8771 033006 105767 002603      TXSETR: TSTB     PGMION      ;PROGRAM I/O MODE?
8772 033012 001407                      BEQ      40$      ;SKIP, IF NOT
8773 033014 032737 000200 173016      BIT      #TXRDY,#TXREAD ;TXRDY BIT SET? **9**
8774 033022 001003                      BNE      40$      ;YES, SO NO NEED TO SET IT AGAIN **9**
8775 033024 052737 000200 173016      BIS      #TXRDY,#TXREAD ;
8776 033032 000207      40$:  RTS      PC
8777
8778                      .DSABL  LSB
8779
8780                      .ENABL  LSB
8781 033034      GOTLIN: ;ENTER HERE WHEN A COMMAND LINE IS INPUTTED.
8782 033034 105267 002556                      INCB     LINGOT      ;SET LINE SYNC FLAG FOR "GETLIN".
8783 033040 103011                      BCC      1$      ;BR IF NO ERROR ON READ.
8784 033042 105167 002550                      COMB     LINGOT      ;NEGATE FLAG TO INDICATE ERROR.
8785 033046 026627 000002 000006      CMP      2(SP),#$TCTC ;TEST FOR CNTL-C, IF SO DISABLE COMMAND REPEAT
8786 033054 001011                      BNE      2$      ;BR IF NOT CONTROL C.
8787 033056 105067 002333      CLRRPT: CLRB     RPTFLG      ;CLEAR REPEAT FLAG.
8788 033062 000406                      BR       2$      ;EXIT.
8789 033064 122767 000130 003327 1$:  CMPB     #'X,TTYBUF+1 ;"X" COMMAND ?
8790 033072 001002                      BNE      2$      ;BR IF NO.
8791 033074 105267 003661                      INCB     XLOFLG      ;SET THE XLOAD FLAG.
8792 033100 000207      2$:  RTS      PC
8793                      .DSABL  LSB
8794
8795                      .SBTTL  EMT DESPATCHER FOR EXTRA EMT CODES.
8796                      ;      NB. NO ROOM FOR FULL DESPATCHER SO ONLY HANDLE CODES 21/22.
8797                      ;      CONTENTS OF R5 MAY BE DESTROYED BY THIS ROUTINE.
8798
8799                      .ENABL  LSB
8800 033102 022700 000042      MOREMT: CMP      #42,R0      ;WAS IT EMT CODE 21 ?
8801 033106 001001                      BNE      10$     ;BR IF NOT.
8803 033110 104006                      EMT      LOADCN
8807 033112 022700 000044      10$:  CMP      #44,R0      ;WAS IT EMT CODE 22 ?
8808 033116 001002                      BNE      20$     ;BRANCH IF NO
8809 033120 005005                      CLR      R5      ;ASSUME NOT CCITT MODEM HANDLER.
8811 033122 000405                      BR       MOREX    ;EXIT WITH R5=0 IF NOT CCITT HANDLER.
8815 033124 022700 000046      20$:  CMP      #46,R0      ;WAS IT EMT CODE 23?
8816 033130 001002                      BNE      30$     ;BRANCH IF NOT
8817 033132 004767 176670                      JSR      PC,CHKSWH ;CHECK KEY SWITCH POSITION
8818                      ;      BR       MOREX    ;EXIT unnecessary branch **9**

```

```
8819      ;
8820      ;note: we donot know who or why this next peice of code was removed
8821 033136 30$::  CMP      #50,R0      ;WAS IT EMT CODE 24?
8822      ;      BNE      MOREX      ;EXIT IF NOT
8823      ;      TSTB     APTLOD      ;LOADED BY APT?
8824      ;      BEQ      40$      ;BRANCH IF NO
8825      ;      BIS      #1,4(SP)    ;SET RETURN C BIT
8826 033136 40$:      ;
8827      ;      BR      MUREX      ;EXIT unnecessary branch      **9**
8828 033136 000167 003250 MOREX:  JMP      BAKOUT
8829      ;
8830      ;      .DSABL  LSB
8831
```

```
8833 .SBTTL CONSOLE TEMPORARY STORAGE
8834 ;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
8835 ;!!!!!!THIS IS START OF R/W STORAGE THAT MUST NEVER BE OVERLAID!!!!!!
8836 ;!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
8837 ;
8839 ;*****
8840 ;THE ABSOLUTE ADDRESSES OF VARIABLES FROM HERE TO END OF CONSOLE(BEGINNING
8841 ;OF PROTOCOL BUFFERS), MUST NOT CHANGE
8842 ;BECAUSE THE ROM-RESIDENT CONSOLE CODE WILL REFERENCE THIS AREA(SEE GLOBAL DECLARATIONS)
8843 ;ANY NEW VARIABLES SHOULD BE ADDED AT THE END
8844 ;*****
8845
8846 033142          FILLTO 35400 ;NOTE: THIS IS A FIXED ADDRESS!!!!!!!!!!!!!!!!!!!!!!
8847
8848 ;DO NOT REORDER.
8849 035400 000020 TCONTL: .WORD INITLD ;TEMPORARY CONTROL DATA
8850          100000          WCSDES=100000 ;WCS FLAG(MUST BE SIGN BIT)
8851          000400          MINSAD=400 ;ADDRESSES UPDATED IN REVERSE DIRECTION
8852          000200          NEGATE=200 ;BIT 7 =NEGATE CONVERSION STRING
8853          000040          DX1FLG=40 ;USE FLOPPY DRIVE 1
8854          000020          INITLD=20 ;SET WHEN CONSOLE BOOTS OR STAR HALTS(NOT VIA HALT COMMAND)
8855
8856 035402 000000 MICFLG: .WORD 0
8857          100000          COMQAL=100000
8858 035404 000000 NEXTCT: .WORD 0 ;COUNT FOR CURRENTLY APPLIED /NEXT QUAL
8859 035406 000000 000000 COUNT: .WORD 0.0 ;USED FOR STEP COUNTS
8860 035412 000 DEEXBY: .BYTE 0
8861 035413 000 DEFSTP: .BYTE 0
8862 035414 000 ABORT: .BYTE 0
8863 035415 000 RPTFLG: .BYTE 0
8864 035416 006416 WHATTODO: .WORD DOSHOW ;WHERE TO GO AFTER PARSING
8865 035420 000 CURRAD: .BYTE HEXRAD
8866 035421 000 CURLNH: .BYTE 0
8867 035422 000 CURADS: .BYTE 0
8868 035423 000 DEFRAD: .BYTE HEXRAD
8869 035424 002 DEFLNH: .BYTE LNGLNH
8870 035425 000 DEFADS: .BYTE PHYSPC
8871 ;.....TO HERE
```

```

8873          .SBTTL  IMPURE AREA FOR DRIVERS AND FILESERVICES
8874
8875          ;READ/WRITE TEMPORARIES AND CONTROL FLAGS
8876
8877
8878          ;CONVERSION TEMPORARIES
8879
8880 035426 000000  SHIFTS: .WORD 0
8881 035430 000000  CNVCNT: .WORD 0
8882 035432 000000  RADIX: .WORD 0
8883 035434 000000  LENGTH: .WORD 0
8884 035436          CONTMP: .BLKW 8.          ;ALLOW UP TO 16 BYTES TO BE CONVERTED
8885 035456          TEMSTR: .BLKW 22.         ;LEAVE ROOM FOR UP TO 44 CONVERTED CHARACTERS
8886 035532 000000  TCTFLG: .WORD 0          ;TERMINAL CONTROL FLAG
8887
8888          ;BIT DEFINITIONS
8889          RSPCFL=20          ;REMOTE TERMINAL SPECIAL CHARACTER FLAG
8890          SPCFLG=40          ;SET WHEN SPECIAL CHAR WRITE IN PROGRESS(LOCAL)
8891          ERRCOD=100         ;
8892          USRREQ=200         ;SET WHEN KBD INPUT REQUEST IN PROGRESS(SIGN BIT)
8893          ROFLAG=400         ;USED FOR RUBOUT SERVICE
8894          REMECH=1000        ;SET WHEN 'TALK MODE ECHO' ENABLED
8905          DISCAR=2000       ;SET WHEN CARRIER ERROR DISABLED
8906          TLKMOD=0
8907          LOCCOP=0
8908          LOCCNT=0
8909          PCARDET=0
8910          PRNINH=100000      ;INHIBIT OUTPUT(CONTROL-0 TOGGLES)MUST BE SIGN BIT
8911          ROUSPR=ROFLAG!USRREQ!PRNINH
8912          REMOPT=REMECH!DISCAR!TLKMOD!LOCCOP!LOCCNT
8913          REMDIS=USRREQ!PRNINH!ROFLAG!LOCCNT
8914
8915          ;DO NOT REORDER, FROM HERE.....TO HERE
8916
8917          KDNVEC: .WORD 0      ;HOLDS DONE VECTOR FOR KBD SERVICE
8918          KUSCNT: .WORD 0      ;USER'S KBD INPUT BYTE COUNT
8919          KBFADD: .WORD 0      ;USR'S KBD INPUT BUFFER POINTER
8920          KBYCNT: .WORD 0      ;POINTER TO KBD INPUT COUNTER
8921
8922          ;.....TO HERE
8923
8924          ;TERMINAL DRIVER TEMPORARIES
8925          SPCCNT: .WORD 0
8926          SPCCHR: .BYTE 0
8927          TERFIL: .BYTE 0      ;TERMINAL FILL COUNT
8928          POSCNT: .WORD 0
8929
8930          FDRV1: .WORD 0        ;USED TO ADJUST SECTOR # DURING DIRECTORY SEARCH
8931          NXTSEG: .WORD 0       ;HOLDS PTR TO NEXT DIR SEGMENT
8932          STRTBL: .WORD 0       ;ACCUMULATES STARTING BLOCK # DURING DIR SEARCHES
8933          SECNUM: .WORD 0       ;HOLDS CURRENT LOG SEC # DURING DIR SEARCH
8934          MESADD: .WORD 0
8935          NOBYTS: .WORD 0
8936          ;DO NOT REARRANGE ORDER OF ECHOSV AND STARCR
8937          ECHOSV: .BYTE 0        ;STORAGE FOR ECHOED CHARACTER
8938          STARCR: .BYTE 0
8939          ;END OF ORDER
8940          FILERR: .WORD 0        ;ERROR CODE FOR DIRECTORY SEARCH ERRORS

```


8941 035572 000000 000000 000000 DIRENT: .WORD 0,0,0,0,0,0,0
035600 000000 000000
035606 000000

8942 035610 000 RXERRO: .BYTE 0 ;FLOPPY DRIVER ERROR FLAG
8943 035611 000 SAVER: .BYTE 0 ;SAVES ERROR CODE FOR 'WAITER' ROUTINE
8944 035612 000 KBDDON: .BYTE 0 ;KEYBOARD DONE SYNC FLAG
8945 035613 000 FRQDON: .BYTE 0 ;FLOPPY DONE SYNC FLAG
8946 035614 000 PRDTON: .BYTE 0 ;PRINTER DONE SYNC FLAG
8947 035615 000 PGMIDM: .BYTE 0 ;PROGRAM I/O MODE FLAG
8948 035616 000 LINGOT: .BYTE 0
8949 035617 000 TIMOUT: .BYTE 0
8950 .EVEN

8951 035620 000000 WAITPT: .WORD 0 ;COMMON RETURN POINTER FOR SERVICE REQUEST EXITS
8952 035622 000000 FLAG: .WORD 0 ;INTER-COMMAND FLAG BITS

8953 ;NOTE: 'SNGINS' MUST BE SIGN BIT
8954 ;

8955 100000 SNGINS=100000 ;SINGLE-INSTRUCTION STEP MODE
8956 040000 IGNORE=40000 ;CLOCK STOP REPORTED
8957 020000 SFWDON=20000 ;SOFTWARE DONE

8958 ; 10000

8959 004000 WCSPRES=4000 ;WCS PRESENT FLAG
8960 002000 USEDEF=2000 ;USE DEFAULT ECO FILE NAME FOR WCS LOAD
8961 001000 SPCSTP=1000 ;SPACE-BAR STEP MODE
8962 000400 SPCSYC=400 ;SPACE-BAR SYNC
8963 000200 INDMOD=200 ;INDIRECT-COMMAND (FILE) MODE
8964 000100 WFDONE=100 ;WAIT-FOR-DONE
8965 000040 SAWERR=40 ;CODE '2' MICRO-ERROR
8966 000020 NOSHOW=20 ;INHIBIT GETTING/SHOWING/TESTING VERSION IN LOAD WCS RTN
8967 000010 QADTYP=10 ;QUADWORD-LENGTH
8968 000004 IDSAVD=4 ;ID-BUS STATE WAS SAVED
8969 000002 SAWHLT=2 ;CPU HALT REPORTED
8970 000001 SECHLF=1 ;SECOND-HALF OF A QUADWORD OPERATION

```

8972 .SBTTL DEVICE REQUEST QUEUES
8973
8974 ;NUMNOD IS NUMBER OF NODES TO BUILD IN LIST
8975 ;SIZNOD IS SIZE OF EACH NODE IN !!WORDS!!
8976 ;FIRST WORD OF EACH NODE BUILT IS POINTER
8977 ;TO NEXT NODE IN LIST.
8989
8990 035624 035626* AVAILP: .WORD AVAIL ;AVAILABLE NODE LIST HEADER
8991 035626 AVAIL: ;NODES
8992 000006 NODSIZ=6 ;6 WORDS PER NODE
8993 035626 BLDLST 16..NODSIZ ;16 NODES OF 'NODSIZ' WORDS EACH
8994
8995 ;NODE OFFSET DEFINITIONS FOR TERMINAL WRITE QUEUE
8996 ;QNXNOD=0
8997 000006 WBFPT=6
8998 000012 WDNVEC=10.
8999 000010 WBTCNT=8.
9000
9001 ;FLOPPY QUEUE NODE OFFSET DEFINITIONS
9002 000002 RXSTSC=2 ;STARTING LOGICAL SECTOR
9003 000004 RXSPFC=4 ;SPECIAL FUNCTION WORD
9004 000006 RXBFAD=6 ;BUFFER ADDRESS POINTER
9005 000010 RXBTCT=8. ;BYTE COUNT
9006 000012 RXDNVC=10. ;DONE VECTOR
9007
9008 036126 000000 RXLQE: .WORD 0 ;LAST NODE IN RX QUEUE
9009 036130 000000 RXCQE: .WORD 0 ;CURRENT NODE IN RX QUEUE
9010 036132 000000 WRTQUE: .WORD 0 ;TERMINAL WRITE QUEUE HEADER
9011
9012 036134 000000 RXTRY: .WORD 0 ;FLOPPY DRIVER RETRY COUNT
9013 036136 000000 INTINT: .WORD 0 ;FLOPPY DRIVER INITIAL INTERRUPT FLAG
9014 036140 000000 RXLSN: .WORD 0 ;FLOPPY DRIVER LOGICAL SECTOR STORAGE
9015 036142 000000 PHYTRK: .WORD 0 ;FLOPPY DRIVER TRACK STORAGE
9016 036144 000000 RXFUN2: .WORD 0 ;FLOPPY DRIVER FUNCTION STORAGE
9017 036146 000000 BYTCNT: .WORD 0 ;FLOPPY DRIVER BYTE COUNT STORAGE
9018 036150 000000 BUFRAD: .WORD 0 ;FLOPPY DRIVER BUFFER ADDRESS STORAGE
9019
9020 ;FLOPPY DRIVER CODE PLACED HERE TO SPEED EMPTY BUFFER FUNCTIONS
9021 036152 105714 TRBYT: TSTB @R4 ;CHECK FOR TR
9022 036154 100376 BPL TRBYT ;BR IF NO TR
9023 036156 000000 EFINST: .WORD 0 ;MOVE INSTRUCTION PLACED HERE
9024 036160 005316 DEC @SP ;DECREASE SHIFT COUNT
9025 036162 003373 BGT TRBYT ;BR IF MORE BYTES TO XFER
9026 036164 000177 140052* JMP @ZFILLP ;RETURN TO DRIVER
9027 036170 000 BOOTFL: .BYTE 0 ;FLAG USED FOR BOOTING STAR
9028 036171 000 TIMFLG: .BYTE 0 ;FLAG USED BY WAIT RTN 'WAITTM'(ROM)
9029 036172 001 NOREMT: .BYTE 1 ;ZERO IF NO REM TERM. MUST BE AT 36172
9030 036173 000 NODRV1: .BYTE 0 ;NON-ZERO WHEN NC FLOPPY DRIVE 1
9031 036174 000 ALLREM: .BYTE 0 ;REMOTE FLOPPY FLAG(SET WHEN ALL FLOPPY REQ TO APT)
9032 036175 000 PASS1: .BYTE 0 ;USED FOR REMOTE FLOPPY OPEN CHECK
9033 .EVEN
9034 036176 000000 000007 FLPTIM: .WORD 0,7 ;FLOPPY POWER-OFF TIMER
9035
9049 036202 036216* CHKFLP: .WORD NOPROT
9050 036204 036216* CHKXMT: .WORD NOPROT

```

9051	036206	036216'	CHKRCV: .WORD	NOPROT	
9052	036210	036216'	OPNCHK: .WORD	NOPROT	
9053	036212	036216'	CHKLCI: .WORD	NOPROT	
9054	036214	036216'	CKXMT1: .WORD	NOPROT	
9058	036216	000207	NOPROT: RTS	PC	
9059					
9060					
9061	036220	036246'	FILLP: .WORD	QUEBGN	;STARLET BUFFER TEMPS
9062	036222	036246'	EMPTY: .WORD	QUEBGN	;STARLET BUFFER EMPTY POINTER
9063	036224	000	QUECNT: .BYTE	0	;STARLET QUEUE CONTENTS COUNTER
9064	036225	000	FLDTFL: .BYTE	0	;FLOPPY DATA FLAG AND COUNT
9065	036226	000000	BUFPNT: .WORD	0	;FLOPPY BUFFER POINTER
9066	036230	000000	KOUNTR: .WORD	0	;FLOPPY BUFFER COUNTER FOR INPUT
9067	036232	000000	FLPFCT: .WORD	0	;FLOPPY FUNCTION VECTOR
9068	036234	000000	DATVEC: .WORD	0	;FLOPPY DATA VECTOR
9069	036236	000000	FLPSTA: .WORD	0	;FLOPPY STATUS FROM LAST FUNCTION
9070	036240	000000	FSECTOR: .WORD	0	;FLOPPY SECTOR
9071	036242	000000	FTRACK: .WORD	0	;FLOPPY TRACK
9072	036244	000000	FLDONE: .WORD	0	;FLOPPY DONE VECTOR
9073	036246	000000	QUEBGN: .WORD	0	;BEGINNING OF STARLET 'RXDB' QUEUE
9074	036250	000000 000000 000000	.WORD	0,0,0,0	
	036256	000000			
9075	036260	000000	QUEEND: .WORD	0	;END OF STARLET RXDB QUEUE
9076					
9077					
9078	036262	000000	RMTQUE: .WORD	0	;REMOTE TERMINAL WRITE QUEUE HEADER
9079	036264	000000	RSPCCN: .WORD	0	;REM TER SPECIAL CHARACTER WRITE COUNTER
9080	036266	000000	RPOSCN: .WORD	0	;REM TER WRITE HEAD POSITION TRACKER
9081	036270	000	RSPCCH: .BYTE	0	;REM TER SPECIAL CHAR STORAGE
9082	036271	000	REMONL: .BYTE	0	;NON-ZERO WHEN WRITE TO REMOTE TERMINAL ONLY
9083	036272	000	PROTOK: .BYTE	0	;NON-ZERO WHEN PROTOCOL ENABLED.
9084	036273	000	SYNC: .BYTE	0	;PROGRAM I/O - LOC/REM SYNC FLAG
9085	036274	000	CSCQTM: .BYTE	0	;USED FOR CONTROL-S TRANSMISSION
9086	036275	000	CTSSNT: .BYTE	0	;NON-ZERO WHEN CONTROL-S HAS BEEN SENT
9087			.EVEN		
9088	036276	000113	CUTOFF: .WORD	LASTOR	

```

9090 .SBTTL RING BUFFER DESCRIPTOR BLOCKS
9091
9092 ;APT PROTOCOL ALTERNATE OUTPUT BUFFER DESCRIPTOR BLOCK
9093 036300 037200 ALTBAS: .WORD ALTBUF ;APT PROTOCOL ALTERNATE OUTPUT BUFFER POINTERS
9094 036302 000176 ALTSIZ: .WORD ALTBFSZ ;SIZE
9095 036304 037200 ALTFIL: .WORD ALTBUF ;FILL POINTER
9096 036306 000000 ALTNUM: .WORD 0 ;# OF BYTES IN ALTERNATE BUFFER
9097 036310 000000 ECHOIN: .WORD 0 ;UNUSED BUT MUST BE HERE(NOW USED FOR ECHO SEQUENCING FLAG)
9098
9099 036312 APTBFO: ;APT OUTPUT BUFFER
9100 036312 037000 .WORD APTBUF
9101 036314 000176 .WORD APBFSZ ;SIZE
9102 036316 037000 .WORD APTBUF ;FILL PNTR
9103 036320 000000 .WORD 0 ;# ITEMS IN BUF
9104 036322 037000 .WORD APTBUF ;EMPTY PNTR
9105
9106 036324 LTEHBF: ;LOCAL TERMINAL ECHO BUFFER
9107 036324 037612 .WORD LECHBUF
9108 036326 000052 .WORD LECSIZ ;SIZE
9109 036330 037612 .WORD LECHBUF ;FILL PNTR
9110 036332 000000 .WORD 0 ;# ITEMS IN BUF
9111 036334 037612 .WORD LECHBUF ;EMPTY POINTER
9112
9113 036336 RTEHBF: ;REMOTE TERMINAL ECHO BUFFER
9114 036336 036624' .WORD RECHBUF ;BASE ADD
9115 036340 000100 .WORD RECSIZ ;SIZE
9116 036342 036624' .WORD RECHBUF ;FILL POINTER
9117 036344 000000 .WORD 0 ;#ITEMS IN BUFFER
9118 036346 036624' .WORD RECHBUF ;EMPTY PNTR
9119
9120 036350 023400' BUF1PT: .WORD BUF1 ;BUFFER POINTER FOR DRIVERS(BUF1 CAN FLOAT)
9121
9122 036352 036414' RMTXPT: .WORD RTIINS
9123 036354 003102' EXTKPT: .WORD RTSINS
9124 036356 003102' WRTMP: .WORD RTSINS
9125 036360 032002' TSTHLP: .WORD TSTHLF
9126 036362 036216' WAITLK: .WORD NOPROT ;A 'HOOK' TO ALLOW SOME CHANGE IN QUEUE BLOCKING SCHEME
9127
9128 036364 036216' NEWCOD: .WORD NOPROT ;EDIT-16 (PARTIAL) : ADD NEW-SELECT-CODES ROUTINE
9129 036366 033102' NEWEMT: .WORD MOREMT ;A 'HOOK' TO ALLOW NEW 'SELECT' CODES
9130 036370 036216' DEADHK: .WORD NOPROT ;A HOOK TO ALLOW NEW EMT CODES
9131
9132 036372 037600 BASEAD: .WORD FRFSIX ;A HOOK TO ALLOW RECOVERING FROM INDEFINITE WAITS
9133
9134 036374 CONRES: ;CONSOLE RESET ROUTINE.
9135 036374 000005 RESET
9136 036376 052777 000100 037756' BIS #RCVINT,@RCR ;RESTORE LOCAL RCV INT ENB.
9137 036404 052777 000100 037762' BIS #XMTINT,@XCSR ;RESTORE LOCAL XMT INT ENB.
9138 036412 012600 BAKOUT: MOV (SP)+,R0
9139 036414 000002 RTIINS: RTI
9140
9141 036416 000000 TTYTMP: .EVEN
9142 036420 TTYBUF: .WORD 0
          .BLKB 82.

```

ZZ-ESKAA-10.1 RING BUFFER DESCRIPTOR BLOCKS
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 92-1
 RING BUFFER DESCRIPTOR BLOCKS

```

9163      000122      TTYSIZ=.-TTYBUF
9164
9165 036542 000000 000000 000000 DATAFR: .WORD 0,0,0,0,0,0,0,0 ;EXTRA WORDS FOR V-BUS CHANNELS
      036550 000000 000000 000000
      036556 000000 000000
9166 036562 000000 000000 000000 DATATO: .WORD 0,0,0,0
      036570 000000
9167 036572 000000 000000 LASADD: .WORD 0,0
9168 036576 000000 SAVIDL: .WORD 0
9169 036600 000000 SAVIDH: .WORD 0
9170 036602 000000 IDTEMP: .WORD 0
9171 036604 000000 000000 EFFADR: .WORD 0,0
9172 036610 000000 000000 GOTID: .WORD 0,0
9173 036614 000000 000000 TBF0SV: .WORD 0,0
9174 036620      000      ALLOC: .BYTE 0 ;REMOTE FLOPPY DISABLE FLAG (ALL REQ TO LOCAL)
9175      .EVEN
9176 036622 032476' RMTVEC: .WORD RMTENT ;NEW REMOTE TERMINAL SERVICE VECTOR
9177
9178      000100 RECSIZ=64. ;REMOTE TERMINAL ECHO BUFFER SIZE
9179 036624 RECHBUF: .BLKB RECSIZ ;REMOTE TERMINAL ECHO BUFFER
9180
9181      ;*****NOTE: DO NOT RE-LOCATE THIS CODE *****
9182      ; EACH OF THESE MUST BE ON WORD BOUNDARY/UPPER BYTE,
9183      ; AS PRESENTLY SET UP.
9184 036724      000 OPBLOK: .BYTE 0 ;OUTPUT BUFFER LOCK
9185 036725      000 MESFLG: .BYTE 0 ;*%* OUTPUT BUFFERS FULL FLAG
9186 036726      000 LOCINT: .BYTE 0 ;SET WHEN CONTROL-C TYPED ON LOCAL TERM
9187 036727      000 DMAERR: .BYTE 0 ;SET WHEN DMA ERROR ON UUT LOAD
9188 036730      000 RFLPWF: .BYTE 0 ;REMOTE FLOPPY REQUEST WAIT FLAG
9189 036731      000 RFLPEF: .BYTE 0 ;REMOTE FLOPPY REQUEST ERROR FLAG
9190      ;*****
9191
9192 036732 022600 DUMBAS: .WORD USRBUF ;POINTER TO BASE OF DUMP BUFFER
9193 036734 000000 DUMPNT: .WORD 0 ;POINTER TO CURRENT FILL BYTE OF DUMP BUFFER
9194 036736      000 MSGNUM: .BYTE 0 ;CONSOLE'S 'NEXT' MESSAGE #
9195 036737      000 ALSTMN: .BYTE 0 ;TEMP HOLDING LAST GOOD MSG REC'VD BY APT
9196 036740      000 LSMSAK: .BYTE 0 ;MSG # OF LAST CONSOLE MSG ACK'ED BY APT
9197 036741      000 MSGLST: .BYTE 0 ;LAST GOOD MSG FROM APT
9198 036742      000 MSGINP: .BYTE 0 ;NON-ZERO WHEN A MSG XMISSION IN PROGRESS
9199 036743      000 SENHDR: .BYTE 0 ;NON-ZERO WHEN HEADER XMISSION IN PROGRESS
9200 036744      000 THDRCN: .BYTE 0 ;# OF HEADER BYTES YET TO SEND(NOT CRC)
9201 036745      000 THCRCC: .BYTE 0 ;# OF CRC BYTES YET TO SEND(1 OR 2 OR 0)
9202 036746      000 000 000 THDRST: .BYTE 0,0,0,0,0,0 ;XMITTER HEADER STRING STORAGE
      036751      000 000 000
9203
9204 036754 000213 INBSIZ: .WORD AINPBZ ;INPUT BUFFER SIZE
9205 036756 037377 INBBAS: .WORD APTBFI ;INPUT BUFFER BASE ADDRESS
9206 036760      000 INBLOK: .BYTE 0 ;INPUT BUFFER LOCK(0 MEANS BUFFER OPEN)
9207
9208 036761      000 XLOFLG: .BYTE 0 ;DOING 'X' BINARY LOAD (INHIBIT DEPOSIT ERRORS)
9209 036762      000 XCMDSV: .BYTE 0 ;'X' COMMAND CHECKSUM GETS PLACED HERE(EDIT 4-07)
9210 036763      000 NOCNLS: .BYTE 0 ;INDICATES CONSOLE SYS OVERLAID PARTIALLY.
9211 036764      000 SETSWH: .BYTE 0 ;FORCES A KEYSWITCH SET-UP WHEN NON-ZERO.
9213
9214 036765      FILLTO 36777

```

ZZ-ESKAA-10.1 RING BUFFER DESCRIPTOR BLOCKS
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 92-2
 RING BUFFER DESCRIPTOR BLOCKS

```

9215
9216 036777      015      TYPE13: .BYTE    13.      ;'ASCII TEXT' MSG TYPE BYTE
9217            037000      APTBUF=.-BASE      ;APTBUF STARTS HERE AND RUNS 192. BYTES
9218            000176      APBFSZ=126.        ;APT OUTPUT BUFFER SIZE
9219            037200      ALTBUF=.-BASE+APBFSZ+2 ;ALTERNATE OUTPUT BUFFER BASE ADDRESS
9220            000176      ALTBZ=APBFSZ        ;ALTERNATE BUFFER MUST BE SAME SIZE AS APTBUF
9221
9222
9223              ;NOTE:*****
9224              ;THE APT INPUT BUFFER MUST START ON AN ODD-BYTE ADDRESS.
9225              ;BECAUSE OF THE 'JMP TO BUFFER' AND 'JSR TO BUFFER' PROTOCOL
9226              ;BLOCK TYPES. SINCE THE MESSAGE TYPE BYTE WILL OCCUPY THE
9227              ;FIRST BYTE OF THE BUFFER, MAKING THE BUFFER BEGIN ON AN
9228              ;ODD ADDRESS PUTS THE FIRST INSTRUCTION IN THE BUFFER ON
9229              ;AN EVEN BYTE BOUNDARY
9230              ;*****
9231            000213      AINPBZ=139.            ;DEFINES INPUT BUFFER SIZE
9232            037377      APTBFI=.-BASE+APBFSZ+ALTBZ+3 ;APT PROTOCOL INPUT BUFFER
9233
9234            037612      LECHBUF=.-BASE+APBFSZ+ALTBZ+AINPBZ+3
9235              ;LOCAL TERMINAL ECHO BUFFER BASE ADDRESS
9236            000052      LECSIZ=42.            ;LOCAL TERMINAL BUFFER SIZE
9237            037664      LECEND = LECHBUF + LECSIZ ;SEE END ADDRESS OF CONSOLE
9238
9239
9241
9242

```

ZZ-ESKAA-10.1 RING BUFFER DESCRIPTOR BLOCKS
V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 93
RING BUFFER DESCRIPTOR BLOCKS

I 12

20-MAY-1986

Fiche 1 Frame 112

Sequence 151

9797

000001

.END

ZZ-ESKAA-10.1 Symbol table
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 93-1
 Symbol table

ABORT 035414R	CANTDO 020662R	CONEQU 022347R	DASH 020712R	DOSTER 004422R
ACTION= 000002	CARDET= 010000	CONERR= 000002	DATAFR 036542R	DOSTPG 004432R
ADDEQU 021454R	CESREG= 000014	CONEXM= 000003	DATATO 036562R	DOSTSO 004454R
ADUPTB 005266R	CGREGE= 000442	CONEXT 017773R	DATEQU 021470R	DOTEST 007210R
AINPBZ= 000213	CHKFLP 036202RG	CONFND 000604R	DATINT= 000040	DOUNJA 003524R
ALLOC 036620R	CHKLCI 036212RG	CONIDN 021563R	DATLST 006756R	DOWAIT 007326R
ALLREM 036174R	CHKRCV 036206R	CONPMP 022414R	DATLTB 005306R	DOWCS 007226R
ALRDHA 020551R	CHKSWH 032026R	CONRES 036374RG	DATRDY= 000002	DOXLOA 014074R
ALSTMN 036737R	CHKSWI= 000023	CONSER 020765R	DATVTC 036234RG	DQAD 021532R
ALTBAS 036300R	CHKXMT 036204RG	CONSPC= 000005	DBY 021513R	DSCLER 013746R
ALTBZ= 000176	CHR?NT 032772R	CONSRT 001150R	DEADHK 036370RG	DSECHO 013770R
ALTBUF= 037200	CKXMT1 036214RG	CONSTR 000624R	DEEXBY 035412R	DSFLOP 014022R
ALTFIL 036304R	CLDSTR= 037746	CONTMP 035436RG	DEEXPM 005022R	DSLOCO 013766R
ALTNUM 036306R	CLEOPA 016600R	CONTSQ 003232R	DEFADS 035425R	DSREMT 014040R
ALTSIZ 036302R	CLEOPL 016572R	CONVER 020464R	DEFLNH 035424R	DSTINT= 100000
APBFSZ= 000176	CLKEQU 021354R	CONVRT= 140022	DEFNAM 017166R	DSTRDY= 002000
APCEQU 021633R	CLKERR 021605R	CONXX 000624R	DEFRAD 035423RG	DSV = 000056
APTBF1= 037377	CLKFAS 021371R	COOVER 017777R	DEFSTP 035413R	DUMBAS 036732R
APTBF0 036312RG	CLKNOR 021364R	COPERF 020004R	DFND 000436R	DUMPNT 036734R
APTBUF= 037000	CLKSER= 140004	COQCLE 020010R	DFOPAC 017006R	DWRD 021520R
APTCMD= 000212	CLKSLO 021376R	COREBO 020014R	DFOPLS 016752R	DXPREI= 140026
APTLOD= 037747	CLKSTD= 000040	COREPE 020020R	DIRENT 035572RG	DX1FLG= 000040
APTRTN= 000366	CLOCKS 021610R	COSSET 020024R	DISCAR= 002000	ECHOIN 036310R
APTSRT 001004R	CLOPAC 016564R	COSHOW 020027R	DISERR 022054R	ECHOSV 035566RG
APTSTR= 000073	CLOPLS 016554R	COSTAR 020032R	DLNG 021525R	ECONAM 017202R
AUTFLG= 037751	CLRRPT 033056R	COTEST 020036R	DMAERR 036727R	EFFADR 036604R
AUTORS= 000004	CLRSIB 003156R	COUNJA 020041R	DNEIE = 000040	EFINST 036156RG
AUTRES 020510R	CLRSND= 020000	COUNT 035406R	DOAUTR 003132R	EINTPE 021130R
AVAIL 035626R	CLSPAR 020710R	COWAIT 020044R	DOBOOT 003104R	ELOCAL 032160R
AVAILP 035624RG	CNVCNT 035430RG	COWCS 020047R	DOCLSO 003210R	ELOCDS 032152R
BADLIN 022355R	CNVERT= 000007	COXLOA 020406R	DOCONT 003220R	ELOCXX 032226R
BAKOUT 036412R	CNVTDN 022505R	CPDBLE 021052R	DODEEX 004514R	EMPTYP 036222RG
BASE = 000000R	COBOOT 017722R	CPHYSE= 000440	DOENDX 006410R	EMTSER= 140016
BASEAD 036372RG	COCLEA 017726R	CPREGE= 000444	DOHALT 003562R	ENCLER 013772R
BITTAB 014012R	COCNTP 020367R	CPTN 021641R	DOINDI 003140R	ENDBLK= 004000
BOOSTR 022472R	COCONT 017732R	CPT3 = 000020	DOINIT 003620R	ENECHO 013744R
BOOTBT= 004000	CODEPO 017736R	CPUIS 021254R	DOIR 006250R	ENFLOP 014032R
BOOTFL 036170RG	CODISA 020377R	CPURES= 010000	DOLINK 013230R	ENLOCN 013740R
BOOTSZ= 000001	COENAB 020371R	CRMES 022430R	Doload 012322R	ENLCCO 013742R
BOOT2 000200R	COEXAM 017743R	CRMESQ 021647R	DONEXT 003730R	ENLTTE 002616R
BOOT3 000422R	COEXDE 006176R	CRXINT= 140012	DOOVER 007240R	ENSEPT 014050R
BOTING 020475R	COHALT 017747R	CSCQTM 036274R	DOPERF 003400R	ENTFOR 032576R
BUFFB = 001000	COHELP 017753R	CSDONE= 000040	DOQCLE 004136R	ENTTLK 032000R
BUFFRP 022542R	COINDI 017756R	CSEBUF= 000002	DORBO 007134R	EOFMES 022405R
BUFPNT 036226RG	COINIT 017757R	CSGO = 000001	DOSHOW 006416R	EOLACT 016526R
BUFRAD 036150RG	COLINK 017763R	CSRD = 000006	DOSHVR 006766R	EOLLST 016520R
BUFO 023200R	COLOAD 017767R	CTSSNT 036275R	DOSSTB 004356R	EREMDS 032250R
BUF1 023400R	COMLOD 007254R	CTXINT= 032560R	DOSSTI 004330R	EREMOT 032270R
BUF1PT 036350RG	COMPAD 004260R	CURADS 035422R	DOSSTN 004414R	ERRCCD 022507R
BYTCNT 036146RG	COMPNX 014660R	CURLNH 035421R	DOSSTS 004376R	ERRCHM 021171R
BYTESL 022013R	COMQAL= 100000 G	CURRAD 035420R	DOSTAR 003420R	ERRCOD= 000100 G
BYTLNH= 000000	COMWAT 003546R	CURRSE 022536R	DOSTCF 004462R	ERRPRG 021203R
BYTSLD 022546R	CONBAS 001174R	CUTOFF 036276RG	DOSTCN 004474R	EXDEPC 006000R
BYTSLF 022544R	CONBOT 001160R	CVNTYP 010436R	DOSTCS 004502R	EXDEV 005232R
CACPAR= 000036	CONCON= 000447	CWAIT 007720R	DOSTDF 007140R	EXECUT 002764R

ZZ-ESKAA-10.1 Symbol table
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 93-2
 Symbol table

EXTENS 022552R	IDCYCL= 100000	LCWRON= 000016	MTCAR1 016314R	MTPROG 015756R
EXTKPT 036354RG	IDDATA= 173010	LDCONS= 000021	MTCLEA 016060R	MTQCLE 015654R
EXTPIO 010414R	IDDATA= 173006	LECEND= 037664	MTCLKP 016066R	MTREBO 016036R
EXUPC 006056R	IDEXDE 005652R	LECHBU= 037612	MTCLOC 016006R	MTRELO 015704R
FDRV1 035552RG	IDMANT= 000200	LECSIZ= 000052	MTCLOP 016014R	MTREMG 016264R
FILENM 022546R	IDNTTB 005250R	LENGTH 035434RG	MTCNTL 016220R	MTREM1 016352R
FILERR 035570RG	IDSAVD= 000004	LINGOT 035616RG	MTCOL0 015742R	MTREPE 015424R
FILLEQ 021476R	IDTABL 010606R	LINKNG 022512R	MTCOL1 016146R	MTSALO 015560R
FILLP 036220RG	IDTEMP 036602R	LNGLNH= 000002	MTCOL2 015712R	MTSAL1 015632R
FILPNT 022554R	IDWRTT= 000100	LNHCOD 022520R	MTCOL3 016170R	MTSAL2 015456R
FILTAB 013702R	ID16 = 000026	LNHDAT 022516R	MTCOM0 015772R	MTSET 015662R
FIRSTW= 010000	IGNORE= 040000	LNKENT 003162R	MTCONT 015610R	MTSHOW 015530R
FLAG 035622RG	ILIEVC 021075R	LNKPMP 022422R	MTCOPY 016226R	MTSOM0 016000R
FLDONE 036244RG	INBBAS 036756R	LOADCN= 000006	MTCOP1 016336R	MTSTAR 015442R
FLDTFL 036225RG	INBLOK 036760R	LOADDE 005412R	MTDEFA 015750R	MTSTEP 015670R
FLNMER 021712R	INBSIZ 036754R	LOCCNT= 000000 G	MTDEPO 015616R	MTSTOP 015676R
FLPERR 021753R	INDBYT 022556R	LOCCOP= 000000 G	MTDFOP 015764R	MTTALK 016204R
FLPFCT 036232RG	INDECH 013504R	LOCINT 036726R	MTDISA 016300R	MTTERM 015726R
FLPSTA 036236RG	INDEXI 022373R	LOCKD = 000001 G	MTDONE 015472R	MTTEST 016022R
FLPTIM 036176RG	INDLFT 022560R	LOCOUT 032140R	MTDX1 016140R	MTUNJA 016044R
FLPYOF= 010000 G	INDLIN 013260R	LODFLG 022514R	MTECHO 016242R	MTVERS 015536R
FMIDHI= 173026 G	INDMOD= 000200	LODMIC= 140042	MTECH1 016306R	MTWAIT 015464R
FMIDLO= 173024 G	INDSEC 022562R	LOISDN 021776R	MTENAB 016176R	MTWCS 016030R
FORCWT 002522R	INEXDE 005434R	LSMSAK 036740R	MTENDX 016162R	MTXLAT 016154R
FPLEQU 022321R	INITLD= 000020	LTEHBF 036324RG	MTEOL 016132R	MTXLOA 016366R
FPLVER= 037755	INITQU 003632R	MAINTR= 002000	MTEQU 015640R	NCAP 020051R
FPVERS= 007600	INITRT 003652R	MAJTRE 015426R	MTERRO 016256R	NCASTK 020362R
FREQ0 = 000010	INSTIV 021032R	MASKS 006166R	MTERR1 016322R	NCBUS 020054R
FREQ1 = 000020	INTIDN 021551R	MATCH 015002R	MTEXAM 015552R	NCBYTE 020057R
FRQDON 035613RG	INTINT 036136RG	MCR = 173032	MTEXIR 015574R	NCCARR 020442R
FRSFIX= 037600	INTR36= 000066	MCS = 173034 G	MTFILL 015734R	NCCLOC 020063R
FSECTO 036240RG	INTSPC= 000003	MDMTYP= 000022	MTFIXA 015602R	NCCMNT 020364R
FTRACK 036242RG	IRIDN 021575R	MEMFAL= 000001	MTFLP1 016234R	NCCNTL 020423R
GEEXDE 005420R	ISANER 021653R	MEMMAN 020737R	MTFLP2 016344R	NCCOLO 020360R
GENIDN 021544R	ISCLR 021314R	MEMSIZ= 040000	MTFPR1 016272R	NCCOMD 020067R
GENSPC= 000002	ISINCO 021672R	MESADD 035562RG	MTFPR2 016360R	NCCOMM 020357R
GETLIN 001600R	ISSET 021310R	MESFLG 036725R	MTHALT 015544R	NCCONS 020073R
GETRNP= 140064	KBDDON 035612RG	MICAST 005442R	MTHelp 016102R	NCCOPY 020432R
GETVER 011424R	KBDINT= 032356R	MICFLG 035402RG	MTINDI 016074R	NCDEFA 020077R
GHMES 022330R	KBFADD 035540RG	MICOPT= 037744	MTINIT 015514R	NCDONE 020103R
GHOPT = 000001	KBYCNT 035542RG	MICWSL 022031R	MTLINK 016116R	NCDX1 020106R
GOTID 036610R	KDNVEC 035534RG	MINSAD= 000400	MTLOAD 016052R	NCECHO 020436R
GOTINP 002646R	KLUDG2 032300R	MMTMOU 021221R	MTLOCA 016212R	NCEO' 020361R
GOTLIN 033034R	KLUDG3 032356R	MNOSIZ= 000006	MTLOC1 016330R	NCEQU 020366R
HELNAM 017174R	KOUNTR 036230RG	MODCHG 032142R	MTNEXT 015500R	NCERRO 020451R
HEXRAD= 000000	KUSCNT 035536RG	MONF 000466R	MTNUM0 015450R	NCFAST 020112R
HLINST 021145R	LASADD 036572R	MOPTFL= 000205	MTNUM1 015506R	NCFILL 020115R
HLTED 021272R	LASADS 022506R	MOREMT 033102R	MTNUM2 015566R	NCFLOP 020120R
HLTINS= 000006	LASDAT 022466R	MOREX 033136R	MTNUM3 015624R	NCFP 020124R
HLTMES 020565R	LASERR= 000013	MOVTOO 002634R	MTNUM4 015646R	NCGENE 020127R
HLTREQ= 100000	LASPOS= 037750 G	MSGINP 036742R	MTNUM5 015720R	NCHEx 020133R
IDAUST= 000040	LASTOR= 000113	MSGIST 036741R	MTNUM6 016374R	NCIDBU 020136R
IDBIDN 021556R	LCANWC= 000014	MSGNUM 036736R	MTNUM7 016402R	NCINST 020142R
IDBSPC= 000004	LGRXVC= 000060	MTBOOT 015522R	MTOVER 016124R	NCINTE 020146R
IDCNTL= 173030	LCTXVC= 000064	MTCARR 016250R	MTPERF 016110R	NCIR 020154R

NCLOCA 020416R	NOREMT 036172RG	QTSALO 016470R	RMRCR= 037766 G	SBIERR= 000031
NCLONG 020157R	NOSHOW= 000020	QTSTAR 016446R	RMRXVC= 000310	SBIUNJ= 000452
NCMNUS 020365R	NOSUFL 021732R	QTTSND 016512R	RMTENT 032476R	SCON 021442R
NCNORM 020163R	NOTREM 022110R	QTHWCS 016440R	RMTQUE 036262RG	SECHLF= 000001
NCOCTA 020167R	NOWCSU 021113R	QUEBCN 036246RG	RMTVEC 036622R	SECLOD 022564R
NCPC 020173R	NRMALL 021347R	QUECNT 036224RG	RMTXPT 036352RG	SECNUM 035560RG
NCPHYS 020201R	NULJOB 002230R	QUEEND 036260RG	RMTXVC= 000314	SECSLF 022540R
NCPLUS 020363R	NULL 014622R	RADEQU 021462R	RMWRON= 000015	SED = 000001
NCPRG 020205R	NUMB1 005304R	RADGET= 000010	RMXBUF= 037774	SENHDR 036743R
NCPSL 020176R	NXTSEG 035554RG	RADIX 035432RG	RMXCSR= 037772 G	SETBYT 017372R
NCQUAD 020211R	ODDADD 013710R	RADLST 006740R	ROFLAG= 000400 G	SETCOM 017416R
NCRELO 020302R	OHEX 021407R	RBUF = 037760 G	ROMBAS= 140000	SETCON 017312R
NCREMO 020456R	OPBLOK 036724R	RCSR = 037756 G	ROMNOP= 000200	SETDX1 017300R
NCR0 020214R	OPENER 013520R	RCVACT= 004000	ROUSPR= 100600 G	SETFIL 017140R
NCR1 020217R	OPENFL= 000003	RCVDON= 000200	RPOSCN 036266R	SETGEN 017320R
NCR10 020252R	OPNCHK 036210RG	RCVINT= 000100	RPTFLG 035415R	SETHEX 017402R
NCR11 020256R	OPNFL1= 000011	RDIDAD 011712R	RSAPV= 140054	SETIDB 017314R
NCR12 020262R	OPNPAR 020704R	RDYIE = 000100	RSPCCH 036270R	SETINP 002736R
NCR13 020266R	OPTMSK= 000001	READ 000220R	RSPCCN 036264R	SETINT 017316R
NCR14 020272R	ORADIX 021403R	READID 011042R	RSPCFL= 000020	SETLAS 005312R
NCR15 020276R	OTHRTF= 140056	READS 000104R	RTCCLR 011034R	SETLNG 017366R
NCR2 020222R	OUTASC 007112R	READSC= 000004	RTEHBF 036336RG	SETLNH 005336R
NCR3 020225R	PASS1 036175RG	REBCON= 140100	RTIINS 036414R	SETLSA 017560R
NCR4 020230R	PCARDE= 000000	RECHBU 036624R	RTIRET 000174R	SETLSD 017704R
NCR5 020233R	PCSEQU 022300R	RECNUM 015106R	RTSINS 003102R	SETMNS 017576R
NCR6 020236R	PCSVR= 037752	RECOG 014326R	RUNBIT= 000400	SETNEX 017410R
NCR7 020241R	PCVERS= 000421	RECSIZ= 000100	RUNNIN 021262R	SETOCT 017400R
NCR8 020244R	PEDT = 000000	RECSTR 014710R	RVSTER= 140074	SETOUT 017672R
NCR9 020247R	PERM = 002000	RELEQU 021505R	SBXBAD= 000006 G	SETPC 017434R
NCSLOW 020306R	PGMIOM 035615RG	RELOCA 022522R	SBTCT= 000010 G	SETPHY 017324R
NCSOMM 020312R	PHEXDE 005360R	REMDIS= 100600	SACQE 036130RG	SETPLS 017574R
NCSP 020316R	PHYIDN 021537R	REMECH= 001000 G	RXC = 177170	SETPSL 017426R
NCSTAT 020320R	PHYSPC= 000000	REMNIP= 036622R	RXDNE = 000200 G	SETQAD 017364R
NCSTEP 020324R	PHYTRK 036142RG	REMLEA 014624R	RXDNCV= 000012 G	SETRPT 017272R
NCTALK 020413R	POSCNT 035550RG	REMONL 036271RG	RXDONE= 173014 G	SETR0 017544R
NCTERM 020330R	PRNINH= 100000	REMOPT= 003000	RXERRO 035610RG	SETR1 017542R
NCVBUS 020334R	PROCD= 000001	REMOT = 000002 G	RXFUN2 036144RG	SETR10 017520R
NCVERS 020340R	PROTOD 036272R	REHLT 010210R	RXLQE 036126RG	SETR11 017516R
NCVIRT 020344R	PRTDON 035614RG	REPLAC 010124R	RXLN 036140RG	SETR12 017514R
NCWCS 020350R	PRTINT= 032300R	REPORT 000404R	RXSPFC= 000004 G	SETR13 017512R
NCWORD 020353R	PSLSTR 021602R	REPOR1 000400R	RXSTSC= 000002 G	SETR14 017510R
NEGATE= 000200	PUSHU 010732R	REQSND= 000004	RXTRY 036134RG	SETR15 017506R
NEWCOD 036364RG	PUTAVP= 140070	RESADD 017332R	R\$SET = 000020	SETR2 017540R
NEWMT 036366RG	PUTRNP= 140066	RESCOM 021013R	R2GRAD 005004R	SETR3 017536R
NEXTCT 035404R	PVER = 000001	RESLSB= 030000	SAVBTE 002512R	SETR4 017534R
NOBYTS 035564RG	QADLNH= 000003	RESMSB= 020000	SAVCOD 022532R	SETR5 017532R
NOCNSL 036763R	QADTYP= 000010	RESNAM 017210R	SAVEFF 022526R	SETR6 017530R
NODRV1 036173RG	QALTRE 016410R	RESTAR= 140000	SAVER 035611RG	SETR7 017526R
NODSIZ= 000006	QTCOLO 016416R	RESTMM 007402R	SAVIDH 036600R	SETR8 017524R
NOECHO 022511R	QTCOL3 016454R	RESTRT 001234R	SAVIDL 036576R	SETR9 017522R
NOLINK= 000005	QTCOMM 016432R	RETRY 000026R	SAWERR= 000040	SETSP 017442R
NOMATC 014776R	QTCOM2 016504R	RFLPEF 036731R	SAWHLT= 000002	SETSWH 036764R
NOOPT = 000000	QTDFOP 016476R	RFLPWF 036730R	SAWTMO 022513R	SETTXR 004444R
NOPROT 036216R	QTNUM0 016424R	RINGDT= 040000	SBC = 000002	SETUPR 017450R
NOREMO 000636R	QTNUM1 016462R	RMRBUF= 037770 G	SBIADD= 000032	SETVBU 017310R

ZZ-ESKAA-10.1 Symbol table
 V10-01-L MACRO V05.03 Friday 25-Apr-86 10:56 Page 93-4
 Symbol table

SETVIR 017322R	SVBOOT 017040R	TMPRAD 022510R	TYP2 = 000013	WHATTO 035416R
SETWCS 017470R	SVBU 021447R	TOIDHI= 173022 G	T1 = 000061	WPMVER= 037753
SETWRD 017370R	SVDEPO 017220R	TOIDLO= 173020 G	T2 = 000062	WRDLNH= 000001
SFWDON= 020000 G	SVER = 000000	TPERRM 013620R	T3 = 000063	WRID12 011006R
SGEN 021425R	SVEXAM 017216R	TRBYT 036152RG	UNKERR 020717R	WRITID 010756R
SHIFTS 035426RG	SVHELP 017154R	TREAD = 000002	UPCEQU 021625R	WRITSC= 000005
SHOWIN 003032R	SVIR 021420R	TRWAIT 000334R	USEDEF= 002000	WRMSTR= 037745
SIDB 021435R	SVLOAD 017250R	TSTCLK 007614R	USRBSZ= 000400	WRTLCP= 140072
SINT 021431R	SWCTIM= 117032	TSTCST 011070R	USRBUF= 022600	WRTQUE 036132RG
SIZTBL 011704R	SYBACT 016670R	TSTERR 007756R	USRREQ= 000200 G	WRTMP 036356RG
SNGINS= 100000	SYBLST 016604R	TSTHAL 010142R	VBEXDE 006050R	WSCVER= 037754
SOFCON= 000017 G	SYNC 036273RG	TSTHLF 032002R	VBUIDN 021570R	X = 036765R
SOMMB = 000100	TAB 021252R	TSTHLP 036360RG	VBUSPC= 000006	XBUF = 037764 G
SOMMIS 021301R	TABMES= 011150	TSTRUN 011124R	VBUSR = 173036	XCMSDV 036762R
SPCCHR 035546RG	TBFOSV 036614R	TSTTMO 011200R	VCLK = 000001	XCSR = 037762 G
SPCCNT 035544RG	TBUF0 = 000022	TSTTY2 010616R	VIEXDE 005364R	XERR1 022146R
SPCFLG= 000040 G	TBUF1 = 000023	TSTVER 011324R	VIRSPC= 000001	XERR2 022157R
SPCLST 015103R	TCONTL 035400R	TTYBUF 036420R	VLOAD = 000002	XERR3 022165R
SPCSTP= 001000	TCTFLG 035532RG	TTYTIZ= 000122	WAITLK 036362RG	XLATFN 011776R
SPCSYC= 000400	TEMSTR 035456RG	TTYTMP 036416R	WAITPT 035620RG	XLOFLG 036761R
SPHY 021413R	TERFIL 035547RG	TWOSPC 020714R	WAITRT 000166R	XMTINT= 000100
STACLS 016544R	TESTLS 015070R	TWRITE= 000001	WBFPT= 000006 G	XXT = 016410R
STARCR 035567RG	TESTND 015006R	TXRDY = 000200 G	WBTCT= 000010 G	Y = 036765
STBOFL 017134R	THCRCC 036745R	TXREAD= 173016 G	WCNEFP 022201R	YESLIN= 000004
STBUS 021336R	THDRCN 036744R	TXRENT 032560R	WCNEPC 022242R	Z = 000012
STCLMP 007336R	THDRST 036746R	TXSETR 033006R	WCSADD= 000042	ZFILLP= 140052
STINST 021331R	TIMEND= 022465R	TYFLER 013644R	WCSDAT= 000043	ZXTMP = 000015
STOPLS 016532R	TIMFLG 036171RG	TYPCAD 011114R	WCSEDS= 100000	\$FER = 000001
STPEQU 021322R	TIMOUT 035617R	TYPEIT= 140046	WCSEQU 022313R	\$FNF = 000002
STRIND= 000400	TIMTRP 022431R	TYPERC 013652R	WCSLOD 020530R	\$FNR = 000003
STRRUN 020601R	TINIT = 000000	TYPE13 036777R	WCSPRE= 004000	\$FOR = 000004
STRTBL 035556RG	TLKMOD= 000000 G	TYPIDR 010464R	WCVERS= 010421	\$TBSY = 000005
STRTCK 004344R	TMEOUT 020640R	TYPTIC 007412R	WDNVEC= 000012 G	\$TCTC = 000006
STS = 000004	TMERTR= 000017	TYP1 = 000012	WFDONE= 000100	\$TER = 000007
STSTA 021342R				

. ABS. 000000 000 (RW,I,GBL,ABS,OVR)
 037000 001 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 12552 Words (50 Pages)
 Size of core pool: 19684 Words (75 Pages)
 Operating system: RSX-11M/PLUS (Under VAX/VMS)

OBJ:ESKAA,LST:ESKAA/-SP=SRC:CONSOLE.801

B 1 Document
 C 1 Document
 D 1 Document
 E 1 Document
 F 1 Table of contents
 G 1 Table of contents
 H 1 V10-01-L
 I 1 **** VAX11/780 CONSOLE(RAM) VER
 J 1 VERSION HISTORY -- EDIT ARCHIVE
 K 1 VERSION HISTORY -- EDIT ARCHIVE
 L 1 VERSION HISTORY -- EDIT ARCHIVE
 M 1 VERSION HISTORY -- EDIT ARCHIVE
 N 1 VERSION HISTORY -- EDIT ARCHIVE
 B 2 VERSION HISTORY -- EDIT ARCHIVE
 C 2 VERSION HISTORY -- EDIT ARCHIVE
 D 2 CONSOLE ASSEMBLY AND LINK NOTES
 E 2 DECLARATIONS AND MACROS
 F 2 DECLARATIONS AND MACROS
 G 2 DECLARATIONS AND MACROS
 H 2 DECLARATIONS AND MACROS
 I 2 DECLARATIONS AND MACROS
 J 2 MACRO DEFINITIONS FOR STAR CONS
 K 2 MACRO DEFINITIONS FOR STAR CONS
 L 2 MACRO DEFINITIONS FOR STAR CONS
 M 2 V10-01-L
 N 2 CONSOLE FLOPPY BOOT
 B 3 CONSOLE FLOPPY BOOT
 C 3 CONSOLE FLOPPY BOOT
 D 3 LOAD CONSOLE PROGRAM
 E 3 LOAD CONSOLE PROGRAM
 F 3 LOAD CONSOLE PROGRAM
 G 3 V10-01-L
 H 3 COMMAND GETTER
 I 3 GET A COMMAND LINE
 J 3 GET A COMMAND LINE
 K 3 GET A COMMAND LINE
 L 3 GET A COMMAND LINE
 M 3 CONSOLE NULL LOOP
 N 3 CONSOLE NULL LOOP
 B 4 CONSOLE NULL LOOP
 C 4 CONSOLE NULL LOOP
 D 4 V10-01-L
 E 4 COMMAND EXECUTION RIN REGISTER
 F 4 BOOT,PROCESS INDIRECT FILE,CLEA
 G 4 BOOT,PROCESS INDIRECT FILE,CLEA
 H 4 START,UNJAM
 I 4 HALT,INITIALIZE
 J 4 NEXT(PERFORM A STEP)
 K 4 NEXT(PERFORM A STEP)
 L 4 QUAD CLEAR
 M 4 SET STEP,CLOCK,SOMM
 N 4 SET STEP,CLOCK,SOMM
 B 5 EXAMINE,DEPOSIT
 C 5 EXAMINE,DEPOSIT
 D 5 EXAMINE,DEPOSIT
 E 5 EXAMINE,DEPOSIT
 F 5 EXAMINE,DEPOSIT
 G 5 MICRO-ASSISTED EXAMINE/DEPOSIT
 H 5 MICRO-ASSISTED EXAMINE/DEPOSIT
 I 5 EXAMINE ID BUS
 J 5 EXAMINE/DEPOSIT STAR PC
 K 5 VBUS EXAMINE
 L 5 VBUS EXAMINE
 M 5 EXAMINE INSTRUCTION REGISTER(IR
 N 5 SHOW CONSOLE STATE
 B 6 SHOW CONSOLE STATE
 C 6 SHOW VERSION INFO
 D 6 SET DEFAULTS
 E 6 LOAD MICRO-DIAGNOSTIC MONITOR 0

F 6 WAIT FOR DONE,SET/CLR MEMORY MA
 G 6 CLOCK TICK REPORTING
 H 6 CHECK FOR CLOCK STOP,WAIT FOR M
 I 6 TEST FOR A MICRO-ROUTINE ERROR
 J 6 TEST FOR A STAR CPU HALT, REPOR
 K 6 TEST FOR A STAR CPU HALT, REPOR
 L 6 TEST FOR A STAR CPU HALT, REPOR
 M 6 TEST FOR A STAR CPU HALT, REP
 N 6 PUSH MICRO-STACK,READ/WRITE ID
 B 7 PUSH MICRO-STACK,READ/WRITE ID
 C 7 TEST FOR STAR CPU RUNNING
 D 7 TEST FOR A MICRO-MACHINE TIME 0
 E 7 PCS,WCS,FPLA VERSION CHECKING
 F 7 PCS,WCS,FPLA VERSION CHECKING
 G 7 PCS,WCS,FPLA VERSION CHECKING
 H 7 READ ID BUS REGISTER ROUTINE
 I 7 FILENAME CONVERSION TO RAD50
 J 7 FILENAME CONVERSION TO RAD50
 K 7 LOAD A FILE
 L 7 LOAD A FILE
 M 7 LOAD A FILE
 N 7 INDIRECT COMMAND LINE RETRIEVER
 B 8 INDIRECT COMMAND LINE RETRIEVER
 C 8 OPEN FILE,TYPE FLOPPY ERROR MES
 D 8 TIMEOUT/ODD ADDRESS TRAP CATCHE
 E 8 TIMEOUT/ODD ADDRESS TRAP CATCHE
 F 8 APT 'X' COMMAND EXECUTION
 G 8 APT 'X' COMMAND EXECUTION
 H 8 V10-01-L
 I 8 V10-01-L
 J 8 PARSE
 K 8 PARSE
 L 8 REMOVE BLANKS,COMPUTE NEXT NODE
 M 8 RECOGNIZE A STRING OF ASCII CHA
 N 8 CHECK FOR A DELIMITER IN INPUT
 B 9 RECOGNIZE AND CONVERT A NUMERIC
 C 9 RECOGNIZE AND CONVERT A NUMERIC
 D 9 RECOGNIZE AND CONVERT A NUMERIC
 E 9 MAIN SYNTAX CHECK TREE
 F 9 MAIN SYNTAX CHECK TREE
 G 9 MAIN SYNTAX CHECK TREE
 H 9 MAIN SYNTAX CHECK TREE
 I 9 QUALIFIER SYNTAX CHECK TREE
 J 9 MAINTREE AND QUALIFIER TREE LIS
 K 9 MAINTREE AND QUALIFIER TREE LIS
 L 9 PARSE ACTION ROUTINES
 M 9 ACTIONS THAT SAVE OPERATION TO
 N 9 ACTIONS THAT SAVE OPERATION TO
 B10 ACTIONS FOR QUALIFIERS AND SET
 C10 SYMBOLIC REGISTER ADDRESS SETUP
 D10 ACT'ONS FOR SYMBOLIC ADDRESSES
 E10 REGOGNITION STRINGS
 F10 REGOGNITION STRINGS
 G10 REGOGNITION STRINGS
 H10 TEXT STRING STORAGE
 I10 TEXT STRING STORAGE
 J10 TEXT STRING STORAGE
 K10 TEMPORARY STORAGE
 L10 TEMPORARY STORAGE
 M10 TEMPORARY STORAGE
 N10 TEMPORARY STORAGE
 B11 TEMPORARY STORAGE
 C11 TEMPORARY STORAGE
 D11 V10-01-L
 E11 CONSOLE SWITCH POSITION CHECKER
 F11 CONSOLE SWITCH MODE CHANGE
 G11 CONSOLE SWITCH MODE CHANGE
 H11 CONSOLE SWITCH MODE CHANGE
 I11 CONSOLE SWITCH MODE CHANGE

J11 CONSOLE SWITCH MODE CHANGE
 K11 CONSOLE SWITCH MODE CHANGE
 L11 CONSOLE SWITCH MODE CHANGE
 M11 EMT DESPATCHER FOR EXTRA EMT CO
 N11 CONSOLE TEMPORARY STORAGE
 B12 IMPURE AREA FOR DRIVERS AND FIL
 C12 IMPURE AREA FOR DRIVERS AND FIL
 D12 DEVICE REQUEST QUEUES
 E12 DEVICE REQUEST QUEUES
 F12 RING BUFFER DESCRIPTOR BLOCKS
 G12 RING BUFFER DESCRIPTOR BLOCKS
 H12 RING BUFFER DESCRIPTOR BLOCKS
 I12 RING BUFFER DESCRIPTOR BLOCKS
 J12 Symbol table
 K12 Symbol table
 L12 Symbol table
 M12 Symbol table